
# Signetics FPLF Oral History Panel

Participants:
Napoleone Cavlan
Ronald Cline

Interviewer:
Andrew Haines

Recorded: September 15, 2009
Mountain View, California

CHM Reference number: X5515.2010

**Andrew (Andy) Haines:** Hi, I'm Andy Haines. We're here at the Computer History Museum in Mountain View, CA on September 15, 2009 to talk about the Signetics field programmable logic family. Here with me are Ron Cline and Napoleone Cavlan. I'd like to start by asking you two to introduce yourselves. Ron, could you tell us a little bit about your background, and the role you played at Signetics?

**Ronald (Ron) Cline:** My background and role at the time was that I was a digital circuit and custom circuit designer designing TTL at the time, bipolar memories, and roughly three years out of school designing [Schottky] bipolar [fuse-link] PROMs [Programmable Read Only Memory].

**Haines:** Where did you go to school?

**Ronald (Ron) Cline:** MIT. I came out of MIT in 1971, three years earlier, with a BSEE.

**Haines:** How did you end up at Signetics?

**Cline:** At that time, Signetics was owned by Corning Glass Works in New York. Besides making glass beakers and stuff they also did passive components. I knew at the time that I wanted to come and work in Silicon Valley,, but in '71 the economy was not that great at the time and nobody from California was interviewing even in Boston. So I interviewed with Corning knowing that I didn't want to work in New York to design resistors. But after the interview, they expressed interest in me coming over there to interview at their plant and I said could they please pass on the interview response to Signetics in Sunnyvale because I knew that they were owned by them and they were nice enough to do that. However, after a telephone interview with my future boss in Sunnyvale, he said, "I'm sorry, we're really interested in you but we don't have the money to fly you out for an interview." I said, "Well, Motorola, who I also interviewed with, will fly me to Phoenix for an interview there. Are you willing to fly me on up from Phoenix to California" and they said yes. So I sort of maneuvered my way into an interview and I got the job for $11,000 a year.

**Haines:** So, Napoleone, tell us about your background.

**Napoleone Cavlan:** Well, I began my engineering career in Boston, Massachusetts. My last job there was at RCA's Computer Division in Marlborough. In 1971, that Division went out of business. At the time I was developing some memories (TTL, ECL, and MOS), and designing with memories, and so when I saw an ad from Fairchild looking for help in this area, I applied. And what happened is that Fairchild flew me here [Mountain View] but at the same time Motorola was also looking for applicants and I had applied there as well. So instead of going through what Ron had, I luckily got Fairchild to pay for my trip to Sunnyvale along with my wife to take a look at the area, and then Motorola paid for a different trip to Arizona where, after comparing the areas for a while, I decided to accept work at Fairchild. Then from Fairchild I migrated to Advanced Memory Systems. From there I went to Signetics because I was looking for a better opportunity and a chance to get into nascent PROM technology. To me, in those days, that was better because I was only involved with dynamic RAMs [Random Access Memories] and static RAMs. So, I was hired in the marketing group as an applications engineer and I started working on bipolar PROMs and eventually migrated to the applications and architecture definition of programmable logic.

  

**Haines:** So you were in the applications side responsible for the applications. Ron you're on the engineering side responsible for engineering the device. So before we dive into the specific effort at Signetics let's talk about the landscape. What was happening in the industry around this around the time that we're talking about here, which is around 1974? So, Napoleone, could you comment on what kinds of IC components were available to designers at the time that you introduced your devices?

**Cavlan:** 1974 was very early to even talk about large integrated structures and to compare them to modern design techniques. There were basically a few tools in the engineer's toolbox that they could choose from to implement logic design. The workhorses were the TTL logic standard gate products and SSI [Small Scale Integration] or MSI [Medium Scale Integration] arrays. TI was the main manufacturer of such products but they were also second sourced, and Signetics made those products as well. Then there were for logic applications 32x8-bit PROMs and 256x4 PROMs. Those were small memory elements that were useful in implementing logic functions in a table format but they had a very limited number of inputs. There were some ASIC [Application Specific Integrated Circuits] products but the ASIC devices at the time were something more akin to the 8008 and 8080 microprocessors and peripheral support chips. [With this moniker I am referring more to the levels of integration and functionality of such products rather than to the uniqueness of the applications they served.] There were some custom products that were developed by individual users in the valley but they were not sold in the public arena. And then, of course, in the area of memories there were ROMs [Read Only Memories], large ROMs but those again were mask-programmed parts that could only be marginally used if at all for logic applications because they were too expensive, and difficult to modify. There was another element that had some modest success, which I used in 1967. These were diode matrices coupled with fusible nichrome [I believe] links manufactured by Harris. They had about 50 diodes per package at the most arranged in vertical row and columns. I used them to design a magnetic core memory simulator for a ROM as a backup for the memory on the Poseidon guidance missile. I also designed a programmer that would program these chips.

**Haines:** So these were kind of precursors to programmable logic devices?

**Cavlan:** Precursors, in that there was really nothing much going on. There was also another part. It was a mask-programmed logic array and this was more like the type of product that we [Signetics] eventually brought to market and which was field programmable. This mask-programmed device was made by National Semiconductor. They called it an MPLA but this basically was all what people could really use to do logic designs other than very specific custom developments for specifically targeted systems.

**Haines:** So I just want to make sure we define our terms here. So MPLA stands for Mask Programmable Logic Array.

**Cavlan:** Mask Programmable Logic Array.

**Haines:** And the PROM stands for Programmable Read Only Memory right?

**Cavlan:** Right. I also mentioned DRAMs, which were Dynamic Random Access Memories.

  

**Haines:** So let's turn to the first Signetics effort here. From the point of view when you were designing this product for launch in 1975, were there other competitive programmable logic devices out there at the time? There was the National mask programmable part.

**Cavlan:** Right at that time, in 1975, the National MPLA device was available and did not enjoy big popularity because of the fact that it was mask-programmed. Often, logic design is not a very structured process. Actually, most of the time isn't, if you're talking about random logic, that's why the name comes to mind – random logic. The inevitable changes that you would incur would not be suitable for using an MPLA, a mask-programmed device. So, the only use you could have really hoped for this part was in more structured logic like look-up tables where you knew more or less in advance what the organization was going to be and the function. But the despite all of that, the MPLA was not successful.

**Cline:** That's why PROMs were usable or were popular as look-up tables — to replace logic because they were able to be reprogrammed or programmed rather and they could change the logic.

**Cavlan:** Right.

**Cline:** Customers love that.

**Cavlan:** Right. That was the impetus. And then in 1975, as we were working to build our own device, Intersil also was working to produce a device and, in fact, their part came out one month ahead of our introduction and then we became the two main providers and competitors in the marketplace.

**Cline:** That particular device was on an older process at the time, a gold doped process instead of a fuse link structure using back to back avalanche diodes. The consequence of that besides, I think basically emulating the MPLA from National with some architecture limitations that Napoleone overcame, it was slower and much more expensive so it came out ahead and caused [us to have] some second thoughts but in the end it did not make it in the market.

**Cavlan:** No, it didn't, the yields were very poor. I remember talking with one of their marketing representatives at the time after we came out with our respective products and they were complaining about a very specific problem. They called it excitation of concurrent product terms [P-Term conflicts] meaning that when the device was addressed with certain logic inputs to it, rather than asserting only those internal gates included in the logic equation of an output for which the logic input was valid, these would cross over and activate another independent logic chain as well, so that the logic output was really confused and was not what they were looking for. They could not eliminate that difficulty, so yield was very low. And I think this was because of the AIM fuse, the Advanced Induced Migration fuse that they had, and eventually the product was not successful.

**Haines:** So that product was not successful. It used a process that was already dated at the time, the gold-doped process. So let's talk about these. What was the Signetics process technology that was available to you?

  

**Cline:** Well, Signetics was very heavy into Schottky TTL, the 7400, 74S line. That's basically what I had been spending my first two years doing —doing several dozen, literally several dozen, family members of 74S logic and that process got to be pretty well manufacture-able. In the meantime, I wasn't part of the team but even when I arrived at Signetics, Signetics had been working on bipolar memories for quite some time. In fact, a couple years before when they were first starting, I believe they were even trying to use aluminum fuses to be the fuse element which didn't work because it took way too much power and the aluminum just sort of went everywhere in bad places. But after that they developed a very robust Ni-chrome process, a Nichrome fuse process and had applied it to the 8223 which is a 256-bit PROM, first gold doped and then a Schottky version of that was selling very well. But they had been working on a 1K PROM already when I arrived and were struggling with getting that to yield. The density was just at the high end. When I was brought into the bipolar memory group in 1973, it was basically to solve that problem of getting a 1K PROM which the company was betting itself on in that division to yield one way or another and that Schottky process as good as it was, was right at its limits.

**Haines:** So you were just a couple of years out of school and this was a key product for the company. How did you get assigned to be the guy that was going to be the white knight to fix everything?

**Cline:** Well I don't think of myself as that that's for sure. I established a track record in the TTL, Schottky TTL and plus I had brought simulation for the first time to Signetics, so I was able to bring some new tools to the circuit designers there within the bipolar memory group to help them try to solve some of these issues. When I say simulation, at that time it was, well we had a program that was not SPICE, it was pre-SPICE called SLIC, standing for Simulator for Linear Integrated Circuits. However, it had a good time domain analysis too. It ran on the 360 business machine. We were only able to run our engineering jobs after the day was finished but there would be batch jobs with card decks and everything. But with just that simulation capability we were able to solve circuit problems that we hadn't done before and we were hoping to bring that technique into the bipolar memory group so that we could figure out a way to make the PROM work. Basically the problem to solve was this – that you had to have a circuit that was able to deliver 40 milliamps or roughly four volts to a 100 ohm fuse in order to blow the fuse reliably, and at the same time have those same transistors be fast enough to meet the spec in terms of speed that was expected by the market so it was a tough job.

**Cavlan:** Well, Ron, if I had known that you were that green I may have had second thoughts about joining Signetics <laughter>. I thought that Signetics really had the ball running on PROMs and had hotshot designers. If I had only known that. If they had told me then, who knows?

**Cline:** By the time I came to the bipolar memory group there were three engineers that had retired on the 1K PROM.

**Cavlan:** Wow.

**Haines:** But you solved the problem and there was a 4K PROM later that set the stage for having the technology being ready for the FPLA [Field Programmable Logic Array], right?

  

**Cline:** Well it set the stage, but also if it weren't for the failures, and maybe there's a lesson here, if it weren't for the failures we wouldn't have had the tools to make the FPLA. And the reason for that is that the 1K PROM was having its main yield problems because they were using an emitter follower array which was a very dense, small area cell but the defect density for that many emitters, transistors basically, in a single array was just too high to get any kind of yield out of that. On the other hand, the 256 bit PROM was a Schottky diode array and it had its limitations from a circuit point of view that at the 256 level you didn't see. At the 1K level it had significant limitations but it could yield and so my job was to figure out a way to get a Schottky diode array to work in a circuit with those current limitations and performance limitations that were needed in a reasonably small die size. And the demonstrator given to me was to make a 4K PROM and to go to the next level. There were some circuit innovations that needed to be made. One was in the column decode which was required at the 4K and 1K level. Column decode had basically an SCR embedded in the decoding which was able to deliver a lot of current but at a much smaller area and that enabled the 4K to work.

Once the 4K yielded, the general manager of the bipolar memory group at that time was Lionel Kirton, who was a great leader. He still is a great man. I happened to see him a couple of weeks ago and he recalled very well the situation that as soon as the 4K yielded he immediately asked me and the design group, the layout group and everybody, all the other engineers, to work on a 1K version of that 4K design because we were way behind in deliverables on the 1K PROM. And in one week we converted the 4K layout to a 1K layout and in another week we had masks and wafers, so in two weeks we went from start of layout to wafers delivered and the final two weeks were the 256 hour burn in [---the qual]. In four weeks, we had a product, 1K PROM product, released to production which was unheard of and still so far as I know still unheard of. Not only that but Lionel, and he still remembers this, when the layout was finished and we were ready to start those wafers to find out if the 1K would work, he didn't start an engineering run. He started 1,000 wafers in production on the bet that this thing would work and luckily this was before I met my future wife and I was able to not sleep on my own but it worked. So then the point being that once we knew that we could manufacture PROMs with a design that worked with both Schottky diode and emitter arrays, we knew that we would be able to have the complexity and the manufacturing capability to be able to make something like an FPLA.

**Haines:** Right and you had the programming technology with the Ni-chrome fuse.

**Cline:** With a nichrome fuse, yes.

**Haines:** That set the stage for creating the device. So let's turn to the product definition. Napoleone, so who is on the team? How did the product get started in terms of the definition?

**Cavlan:** The way the product got started was really in design engineering because at the time I was involved in memory applications and my mind wasn't involved in defining products. Defining products for the division was not my duty or my responsibility. I was familiar with mask-programmed arrays that were offered by National, not Intersil. Intersil's part wasn't even in view yet because it had not been announced, but I knew what those products were. And one day I suppose the way it happened, if I remember well, I was basically presented with the fact that we wanted to do this and it was a straw man structure that was being designed and there were some decisions to be made, and that's where at that point I can call that the formation of a team because now I got involved. Design was involved and then, of

  

course, marketing had to be involved because they wanted to know whether this was a product that could succeed. So if you want to talk about what the team was and the number of people involved, it was basically myself, Ron, and some product engineering representative because they were eventually going to put the product out, and then marketing was the decision maker along with the final verdict from the division manager. But in terms of the team it was really, I would say, at most four guys. That's it.

**Cline:** That actually implemented it.

**Cavlan:** Yes.

**Cline:** Now I should mention that there had been a background. John Campbell, who was the design manager when I came into Signetics, told me just recently that the concept had been floating around or had been talked about by customers every once in awhile that they would like something like that. The big iron companies at the time knew what PLAs were. They used mask programmable PLAs as micro code or micro instruction generators inside of their CPUs but, again, because what I was just telling you earlier there was just never any consideration of the possibility. Again, after the 4K [PROM] worked, it was at about this time that Napoleone came on. I recall, I didn't know anything at the time. Nobody had told me. They didn't want to defocus me about this other idea that was floating around but I found on my desk a request from an engineer from Honeywell, one of the big iron companies at the time, "Could we make a fusable or a fused link PLA?" That was the first time I'd ever seen the concept but, again, it was something that just I think the technology capability, the circuit capability, the applications and knowledge, plus a built-in recognition of customer desire I think just sort of came around at all the same time.

**Haines:** So it all came together so the computer companies were interested. You had the technology. Napoleone, who asked you to start focusing on this? Do you recall?

**Cavlan:** Well it was the marketing group. I was in charge of putting the device together in terms of a data sheet, any kind of application notes and any kind of marketing flyers that eventually would go out and provide support to our users. So it was my marketing manager at the time, whom I believe was Ralph Kaplan. He was the marketing manager. So the directive is – OK, you're supporting PROMs but now you have to dedicate some of your time to this.

**Haines:** And so what was the process that you used to start to define that architecture?

**Cavlan:** Well the process to define the architecture was not a usual process that you might envision occurring today, nor that occurred later where there was a more organized way of addressing market requirements. The idea was that this was a device that seemed to have a lot of general market appeal because of the success of the mask programmed devices and the desirability by people to modify logic in their design environment. So to translate that intent into the architecture was basically saying, "Okay, well there is a mask-programmed version available. Let's make a field-programmable version, and make one that can be targeted to a broad range of applications that are not just bit oriented, also byte oriented." And in that respect, I took it upon myself to take a look in these two areas, and I was convinced that in architecting a device that has a set of programmable AND gates and programmable OR gates but with an I/O combination which was basically byte organized (in our particular case it was 16 inputs and 8 outputs)

  

with an enable for the output gating, it would be useful not only for generating bit logic slices out of the device, but you could use it as a memory element where you put its entire output as a byte on a bus and you enable it in a tri-state configuration. And that was the thing that I fought for in the definition of these products and so that's how the device emerged. There was another issue regarding the number of AND gates that we were going to put on the device, which were referred to in those days as product terms. These were the columns, if you look at the geometry of the die, and we were limited with the technology to a certain die width because it had to fit in the 28-pin package. It's the only package we had. And so the most we could do was about 48 product terms and you <turning to Cline> mentioned something that was a motivation for that, it was about…

**Cline:** It was half the National [Semiconductor] part.

**Cavlan:** Yes, half the National part, and also something about the number of products required to implement a Hollerith code.

**Cline:** Well the 96 for the National part was the Hollerith code, which was a strange number. They probably did that because they couldn't do 128 or something, but anyway that's historical. But you fought with me for quite a while before you finally accepted the 48. I remember that.

**Cavlan:** Yeah. I wanted to go to 64. I didn't like the number 48. It didn't really make sense to me; it is not an even power of two.

**Cline:** We were scratching for every square millimeter we could.

**Haines:** So you said you had to fight to get the byte oriented architecture accepted. What was the fight about?

**Cavlan:** Well the fight was that there was a competitive, an alternative architecture proposed. The alternative was very simple, dull one, in my mind at the time: just copy the National part. And to me, I never wanted to copy anything and, in fact, as the product evolved you can see why <laughs>. It was like making American cars and making European cars, kind of different concepts in terms of style and whatever. But regardless of that analogy, I just was reluctant to just do a copy and also saw there were advantages in doing it our way because the product would be more useful. And so that was the reason why I really pressed, and eventually I was able to win the argument.

**Cline:** You had the two extra inputs, the full two bytes wide plus the two chip enable switch also gave me as a circuit design point of view a degree of freedom.

**Cavlan:** One chip enable.

**Cline:** Pardon?

  

**Cavlan:** One chip enable.

**Cline:** One chip enable. Well that was all I needed then for the fuse supply.

**Cavlan:** That's right. [One pin for logic output enable, and another dedicated to fusing voltage supply].

**Cline:** So to drive home the point though about the fight for chip area, I mentioned earlier that the technology that Signetics had was really a key aspect of it, meaning they could do Schottky arrays but also at least at a smaller density could do emitter arrays. If somebody was going to design this, a PLA, normally they would have the AND array, then de-Morganize the OR array following it with an inversion and then a NAND array, so basically the same element in both arrays but that required then another set of buffers and decoders between the arrays. When I looked at that, it just would have blown up the die size and made from a power area---everything---point of view it just wasn't makeable. But because of the Signetics technology and our history and knowledge, we were able to have the AND array formed by a Schottky diode array going up--- a true positive "AND"--- and then the OR array without any inversion in between, the OR array coming back down through an emitter follower array and generating an OR directly with one single resistor pull up on each product term. That simplified the design considerably and made the die size reasonable and, again, we were able to manufacture and yield that so that was really a key aspect to the experience.

**Haines:** So you talked about minimizing the size. Did you have specific performance, power, cost targets?

**Cline:** Let's see. The size was limited by the package.

**Cavlan:** Right.

**Cline:** The power was limited by the package basically.

**Cavlan:** 180 mils is the most we could put on it.

**Cline:** Exactly, so I mean it was basically a one watt device and, by the way, the exact size is about 12 square millimeters, one watt, that's about ten watts per square centimeter and that's been a constant in programmable logic for 35 years based on the requirement of 500 linear feet per minute as cooling, the cheapest cooling. This is a commodity product. Customers expect to be able to cool it and programmable logic is always operating right at the performance limit and no matter what the technology, whether it's TTL, ECL, CMOS or whatever comes next, it's 10 watts per square centimeter. But anyway, that was the power and then the performance was just as fast as we could do it.

**Cavlan:** We made 50 nanoseconds, which was faster than the Intersil device I think.

  

**Cline:** Intersil's was 90.

**Haines:** How did that compare with the TTL kind of stuff that was around?

**Cavlan:** Well, it was slower. If you took a TTL package, a TTL gate package in those days was about 10, 15 nanoseconds. So it was not as fast but the idea was that by consolidating logic… see the architecture of our first FPLA (82S100, PLS100) was basically an AND-OR/NOR structure because there was a polarity inversion on the outputs, I mean internally, so you could do AND-OR or AND-NOR functions. But it had a very large input width so that you could take 16 variables at the same time and put them into any one gate. Not only that but you could take a number of AND gates and distribute them at will among the outputs and also you could share them. So when you looked at the logic function done with TTL, usually by manipulating all those inputs, even for a single slice, you ended up with several levels of TTL packages. When you looked at that chain, the delay was sometimes much greater than just one FPLA. Sometimes the delay was smaller. It was a faster chain with TTL. In those cases, the FPLA was not the best solution, but if speed was not critical you still had the advantage of putting all those TTL chips in one package into the FPLA, one single package, and that was very useful.

**Haines:** The package sounds like it was really very important because it defined the power. It defined the number of pins and the die size that you could put in the package. So a lot of your architecture – what was possible – depended on your choice of package.

**Cavlan:** Right. Because at that point, well… but you have to look at another angle too. We could have constructed a device that fit in a smaller package but then we had to accept the consequences and impact on the architecture of these parts that we wanted. In those days, the drive was to provide as much logic density as we could because of what was the first impression of the market, really what it was looking for, from the seed that had been established by the National part and the desire to integrate. So the denser the FPLA was, the more its integration and cost effectiveness. It was only later that the market developed in a multi-pronged facet, which required different types of devices. We can talk about that too.

**Haines:** Okay. So the architecture was defined and you're ready to go. What was the signoff process? What was the management approval process like?

**Cavlan:** <laughter> It was a very informal affair if you compare it to the modern designs and the modern product development processes that go on in large companies where the investment is so large that a lot of people from the lowest level to the highest level get involved in the selection process. However, our methodologies were very simple. Once the engineering and the application people were agreed upon design feasibility from one side and desirability from the other side (because it was more or less a vision of what the market would take), then it was just an issue of convincing the Division manager saying, "Okay, this is what we're going to do." And usually the Division managers that we had at that time…well Lionel, I don't know. He was more like a process guy, wasn't he?

**Cline:** Yes.

  

**Cavlan:** So he had a lot of trust in the groups that managed that aspect, the application marketing and design. And later on we migrated into managers that were almost entirely business-like, so they were really spending their time in the office. As long as you presented the product plan and showed that you would make some money that was it. They went for it, and that was the process. So in our case with the FPLA, it wasn't difficult at all. Once they got assured from Ron that he could make the part and was assured from us, "Yeah I think that there is some"…

**Cline:** Well first I had to convince my boss at the time, Dale Leuthold, who was the design manager at that time, very well respected. He was an ECL designer and just a super circuits guru. First I had to convince him that the approach that we had proposed for the arrays, et cetera, would work. But then the other person that I had to convince was Mike Hackworth who at the time he was either the GM or the senior marketing manager.

**Cavlan:** He was the Division marketing manager.

**Cline:** Division marketing manager and I remember him calling me into his office at the time and sitting me down and saying, "I'm about to decide whether to make a bet on whether we go forward with this and it's going to involve a huge investment for this division" and basically he asked me to look him in the eye and then said, "Is this something that we can do?" So I said yes I believed that it was and it made sense but that was the signoff process.

**Haines:** So you got the approval so you're off and running. So let's talk about the chip design process then. So, Ron, what was the bipolar technology you had at the time to work with?

**Cline:** Beyond just being just the general Schottky transistor process, in 1974-75 it was a two micron, basically two, two and a half micron lithography. The smallest contact was 2x5 microns. The smallest transistor then, bipolar transistor, was something like 20 microns by 30 microns. In order to put that in perspective, a two micron lithography, right now my design team are working out with now we're looking towards the next generation. It's roughly 100 times linearly dimensioned smaller, 100 times linear is 10,000 in area. That means, in area, in the size of one transistor at that time we could put roughly 10,000 transistors today and I double-checked that number yesterday and that's really the truth. So understanding that context at the time you were not able to have a lot of transistors on a chip. I think the FPLA ended up being somewhere around 1,000 or 2,000 or a few thousand transistors at the most. The technology was again TTL so there were resistors. The resistors were even larger than the transistors but we were able to-- Signetics had one of the very first dual layer metal, two layer metal technologies and it turned out that that process in order to get again the currents that are required even though it was on a 10 micron pitch was also an enabler of this. That's basically what we were at then.

**Haines:** So, Ron, what kind of computer resources did you have in this design process to help you?

**Cline:** Well, as I think I mentioned earlier in those days the only computer that was around for engineering work was the mainframe business computer at the plant. There was a small VAX machine but it was used only for helping to work with the layout data, which I guess I'll describe a little bit later. But basically everything is by hand. The schematics were drawn by hand. The 82S100, the FPLA here,

  

was drawn, the schematic was on one B sized sheet of paper, clear print drawing, easy to erase. And in those days we didn't even have a copier that shrank so the schematic was always copied and then taped together on two 8-1/2 x 11 inch. There were a lot of those versions of the copies of the schematic going around even in product and test engineering so that was done by hand. The layout was done by hand on drafting tables. The layout was then converted to card punched vectors, if you will, line by line so the FPLA design ended up being a box of IBM cards about yay by yay [6 inches by a foot] and woe be the person who spilled that box of cards. I mentioned that we did not have— it was still drafting and it was not done with any Calma system. We didn't have that yet. This was one of the last products that was designed that way but it was one of the first products, if I remember right, at Signetics where it was not done without a Ruby. Before that even our biggest products they were cut on Rubylith which is that famous laminated sheet of red plastic that would be stripped away rectangle by rectangle and then it would be put up on a wall that was about 20 feet away and then a big reduction camera would take a picture of it and you would make masks out of that. This product was one of the first to use I believe it was a PerkinElmer, what was called a pattern generator and it converted those IBM decks of cards through this computer, this small computer, IBM 1104 I think and converted it into flashes of rectangles and the reticle, the 20X reticle was exposed directly that way and that was a big change. That wasn't part of the design process so the design was still done by hand and checked by hand. The drawings were generated with CalComp plotters and layer-by-layer. Although I was the prime engineer there were a lot of second engineers, other engineers, who helped me ensure that everything was hooked up correctly by just manually going through all those plots.

**Haines:** So how many people were there on the team then? So you were the principal designer. How many people did you have helping you?

**Cline:** Just on a helpful basis, just one or two other engineers. These were the guys who were still at that time working on other PROMs. They were the lead engineers on that, Dan Medler being one, and then Dale, my boss, would come in and help every once in a while too. We didn't have a big team.

**Cavlan:** Funny you should mention Dan Medler. Do you know why? Because he ended up buying my house, my own house, yeah in what was it: 1976? I put my house on the market for sale and so who shows up: <laughs> it's Dan Medler and his wife buying it, so that's how we did it.

**Cline:** That's interesting. There was a concern by I would say some of the engineering management, those above me, still of was this going to really get pulled off? And so John Campbell, who was the one who hired me out of school, suggested to Dale that maybe we should pull some engineering resource from the old days and so he located Sam Sirkin who had gone. He had retired by the time I got to Signetics in '71 but Sam was an old time circuit designer. He was a seat of the pants circuit designer literally because he came out of the navy as a carrier pilot, then got his engineering degree, came to Silicon Valley---t wasn't "Silicon Valley" at the time---designed a whole bunch of circuits for Signetics including an 8-bit static RAM in 1966 if I remember right. But I had never met him but he was found in Mexico making candles and John brought him up to Sunnyvale for a week to review my design, work with me on it, and I'd have to say Sam made some significant contributions. It would have been a 70 nanosecond part instead of a 50 nanosecond part if Sam hadn't figured out a way to make that [sense amp] a little bit faster. I'll give Sam the credit for that.

  

**Cavlan:** By the way, I want to interject another thing. I just remembered that I made an error in my mind. It was not Dan Medler. It was another guy who was a product engineer, and right now the name is escaping me and I'm desperate to try to correct this, but it was not Dan Medler so for the record.

**Haines:** So you were thinking about that.

**Cavlan:** Yeah, I was thinking about it but I confused the name; but that's okay.

**Cline:** But that reminds me though of something else. When he said product engineer, design teams were very small so, yes, I was really the only circuit designer on it but if it weren't for the product engineers and testing engineers that went along with that, they were really what made the product manufacture-able and working with them. George Connor was the head of the product and test engineering team and he and his people really were deeply involved in the design, making sure that they could test it.

**Cavlan:** Yeah, I remember. I remember George. And, by the way, I just remembered now who that person was, and you can see the confusion. It was Bob Gretler. Bob, you remember him?

**Cline:** I remember Bob, yeah.

**Cavlan:** You see… Dan Medler, Gretler, I was a little bit, … okay.

**Cline:** Wow.

**Cavlan:** He did, yeah.

**Cline:** Yeah, okay.

**Cavlan:** And that's for sure.

**Haines:** So what were some of the biggest challenges and innovations that you had to make to get this device done?

**Cline:** That OR array being the emitter array that was always the bane of the PROMs but somehow we had to make it work and somehow to deliver the power to that fuse going through the product terms and also be fast – that was a huge challenge. But even more than that, and this is where I really threw the ball into Napoleone's court, and I know you know this is to do the row decodes for both the AND array and the OR. I knew I had to have a product decode so that was just a given. It was a waste and that's always when you're doing programmable logic that's always a concern because you know at the end that the data path itself after it's programmed all that's left is typically maybe about 30 to 35 percent of the

  

whole chip area. The rest of it is used once and it's thrown away. So you're really concerned whenever you have to add anything. So the product decode I knew I had to add but if adding row decode, PLA is not like a PROM. It does not have a built in decode in the rows so we had to add it somehow. I talked Napoleone out of having a row decode to address the rows in order to program it. Instead, I came up with this Rube Goldberg approach. What if on all the inputs, if you raise an input to a super voltage instead of five volts we raised it to eight volts, it disables that row completely so the true and the complement are both off. The row that we want to program we let it be at either a low or a high logic level, zero or five volts, and so we programmed that row, the column decode, programmed that fuse, switched the row, and then on each row at a time, lower it from its super voltage, give it a high or a low so it's, what, literally a linear select and it required all the inputs to do that. But it also required a very complex and very unwieldy programmer and programming algorithm to do it but if it weren't for that to be able to do that we would not have been able to make the die size at all.

**Cavlan:** Yeah, it wouldn't fit in the width.

**Cline:** Yeah.

**Cavlan:** And the part that he's addressing with that is the fact that there were no decoders on the horizontal rows, and because of a problem in manufacturing the programmer or architecting the programmer. Because the programmers those days essentially were being modified… they were extensions of PROM programmers, and in those days I think it was the Model 5, if I'm correct, from Data I/O that was one of the workhorses in the industry. And they had a bunch of lights and switches on them trying to address the PROM. That was very unwieldy and I was wondering whether we would have to go to another situation like that for an FPLA programmer to have all these switches for selecting the rows and the columns. But fortunately we were able to avoid that by going to another methodology of keying in the information on the first FPLA programmer that Data I/O manufactured.

**Haines:** So you were worried that innovation in engineering was going to cause a problem for the customers and the programming but it sounds like you got a work around.

**Cavlan:** Right, well eventually I got together with Data I/O in defining the programmer the way we envisioned it would work because there were some issues that were related to the program table being used in support of programming the product from my user point of view. And so there were a number of considerations. First of all, we had to use a display, a CRT display to handle the data; and then when I say we… Data I/O had to do it but I was working, collaborating with them on the phone all the time with the main design manager. That was Milton Zeutschel, and I called him constantly over the phone. At one point, they thought maybe I was being a pain in the butt trying to steer them in certain areas. One of these in particular was entering the characters for programming, like using a keyboard on a calculator rather than having switches or push buttons. So, eventually they responded quite well to that and had their own ideas how to do that. So, they eventually made a useful machine, which turned out however to be very expensive. But at any rate, they were able to handle the programming algorithm that Ron had defined, and so we were lucky.

**Haines:** So the device turned out to be it was 5,000 to 10,000 transistors is that right?

  

**Cline:** Yeah.

**Haines:** And 12 square millimeters was the final size. How much is that in mils?

**Cline:** One twenty by 150 so whatever that is.

**Haines:** Okay, in mils. That's what was used back then: mils. So what was the time period for this? How long did it take to do the architecture definition? What was that?

**Cavlan:** The architecture was basically a brief thing because basically the layout was there of what the FPLA should look like based on the National part. The debate was whether we should extend it, put it in a 28-pin package, 48 product terms, more output pins, and an output enable, and so forth. And I think it didn't take long to resolve that one. I don't think so.

**Cline:** No.

**Cavlan:** Yeah, maybe a month or so where it went back and forth because every time it was a push back from engineering saying, "Well, the die gets too big" and then the cost of the product increases, plus the difficulty in designing it. Eventually we went to… <laughs> I think it was a meeting with marketing. You and I <turning to Ron> went there and we hashed it out and finally there was a decision. <laughter> So…okay…they went for my side. I think it was the right thing to do.

**Cline:** Oh yeah, absolutely.

**Haines:** So it was about a month and then so you had a spec then. So how long did it take you to go from that to masks?

**Cline:** I honestly don't recall but if we brought it out in 1975 my guess would have been about two to three months. Considering the tools that we had and everything was done by hand, we had a— well something about that day and age that people don't appreciate was that the layout designers were as important to the process as anybody else, the engineers and everybody else who was degreed. Layout designers typically were housewives, sometimes empty nesters but generally women in their 30s and 40s on average who had a certain combination of skills of art and math and perspective, problem solving that was totally— Silicon Valley wouldn't exist without them. Let me digress. When I first came to Signetics in 1971, my boss then, Lu Hintz, who later went to Intel, but he gave me the job of designing a simple two input NAND gate. It was on a gold-doped process at the time. It was a 7420 or something like that. He had me figure out the transistor sizes and there was basically a cookbook at the time. And then he said and I was just in my design room he said, "Okay, the next thing you got to do is you got to lay it out." He didn't tell me that there was this room full of draft layout designers down at the other end of the building. I hadn't walked down there yet. I had only been there a couple of weeks or so. So he gave me a day to do that layout. Now in those days it was a one layer metal process. I decided that he had given me, just to be funny, an impossible task after a day. I couldn't figure out a way to hook it up in one layer of metal,

  

pad input to output. So I said, “Lu, I’m sorry you’re pulling my leg.” And he said, “No, why don’t you give it another day.” So I worked on it a second day. I couldn’t figure it out. I could not get this thing so it would layout on a piece of silicon. So he took me by— I went back to him and he took me by the shoulder and walked me down to this other end where there was a roomful of these ladies and he took me over to Jean Degrenier, who eventually did the FPLA, and introduced me to her. She took the schematic and smiled at me in that friendly way of hers and an hour later she had the finished layout for me. And so that was a bit of perspective on what it takes in order to make a real integrated circuit. There are no heroes. There are just a lot of hardworking people with their own skills, their own talents, and it took everybody to be able to make this thing work.

**Haines:** That’s a great story. So it was a number of months and so you’re ready for fabrication. So how did the handoff happen to fabrication and where was it fab-ed and can you give us some of those pieces?

**Cline:** So after the masks were made there locally at Signetics, Signetics at that time had its own mask shop. It made its own masks and the masks were carried 100 yards to the fab, all of this at 811 Arques, Wolfe Road and Arques, where the main plant was and the fabrication probably took about, my guess, about three or four weeks to go through the fab to make that part.

**Haines:** And then how was it tested? What was the process for testing these devices?

**Cline:** Oh, the methodology that we used back then other than just straight logic was on a Teradyne tester but the methodology since it is a nichrome fuse link we actually had to have extra rows and columns in the design.

**Cavlan:** Yes, you put those in.

**Cline:** So it was really a 50 p-term part.

**Cavlan:** That’s right.

**Cline:** But you had to throw away two of them to get what you wanted. And they were two rows and two columns in order to be able to generate a pseudo checkerboard pattern in those test rows and columns so that they could make sure that there weren’t any shorts that were generated that they couldn’t detect. But so every other fuse was blown on each of those test rows and columns to make sure that it was programmable.

**Cavlan:** Right that was for testing and then eventually those two, we called it the test array, those two product terms that would be eliminated by the customer programming cycle in the Data I/O machine. We just tossed them away because they would not interfere with logic. But when they were shipped to a customer at least he could test the part and make sure that the output toggled.

**Cline:** Yeah. That’s right. It was a functional test for them too, yeah.

  

**Cavlan:** Right.

**Haines:** Right, so you built in a little extra redundancy explicitly for testing because it's a one time programmable.

**Cline:** Yeah and that was an important part and I think that raises a good point is that since we ship them without a pattern and in the end the pattern, the function is completely undetermined. If the customer wanted to test the product without this capability they'd have to develop their own vectors based on their own pattern and that would have been a big expense. So at incoming test inspection because we had it this way they could test that the device worked before they programmed it, yeah.

**Cavlan:** Right. That was a primitive way of saying functionally the part is okay. There are no shorts between rows and columns that in certain areas may… at least no shorts in the rows. You couldn't predict, when you programmed the rest of the arrays, that there were other problems that would surface. But at least the manufacturer knew that the product was alive, because for a virgin product with all the fuses intact you couldn't tell anything at all. And if you programmed it in any way you would be stuck to using your own code, and then you couldn't return it because it was already programmed and would get into an argument with the factory to resolve. So the test array was a good idea to put in.

**Haines:** So how did they yield when they came? You started to get devices back. Were the yields where you wanted them to be?

**Cline:** Actually other than a couple of design quirks which I've blissfully forgotten, I'm sure it took a couple of iterations, but the yield wasn't bad considering the die size that it was but it was still an expensive part and that was really the challenge. We knew from the get go because it was not going to be a cheap part and that really was— how do you sell a $25.00 part?

**Cavlan:** Right, we introduced the product with very high ASPs. I mean in small quantities you'd pay about $35 for it and in volume, we went down to about $12...

**Cline:** But those are 1975 dollars.

**Cavlan:** That's right. But we made sure that we got good margins because that was one of the ideas, that with such a proprietary product we would be able to increase the profits of the Division. It was a profitable product while the ASPs, the Average Selling Price of PROMs was declining, especially in the low densities; and also we could leverage the product. So we maintained a high selling price for a long time, but eventually it went down to four or five dollars.

**Cline:** Yeah. I don't recall there ever being a manufacturability problem with this, at least on a scale or in comparison to the PROMs that went before.

**Cavlan:** No, it was a good yielding device and also readily available. That's the other part...

  

**Cline:** Yeah. And that was important.

**Cavlan:** Yeah.

**Cline:** But, even then, I don't know if the question is ready yet but, in terms of customers in order to be able to get the design in wanted a low price alternative and it wasn't long before some high visibility customers, DEC and IBM, as I recall said, you know, "This is a great product. We can see it's available, it just doesn't meet our long-term cost goals. We've got to have a cost reduction path," and there were no ASICs at that time or at least as part of a natural flow as we would understand programmable logic, to have an ASIC cost reduction path now. The only possibility was to have a ROM or masked ROM version of it to be able to lower the cost potentially and we debated this for awhile. But eventually, design was asked to very quickly come out with a masked version of the FPLA and we did. We cut that die size by over a half, it was about one-third the size. Of course, it didn't need the fuse links, so it saved some process cost and we got that out. It was programmed using the second layer metal, the last metal programmed, and the test engineers, probably Bob Gretler, in fact, I think it was Bob, because you mentioned that name and we're starting to bring back memories, came up with an algorithm to take the code and our mask information and be able to generate rectangles in just the right spot in the metal to create the pattern. It worked first time out and went right into production. We released to production. It went into the data book. We got those designs from DEC and IBM but, as far as I remember, we never sold a single one but it was required in order to get the design.

**Cavlan:** It was a dream part for you, Ron. Now I know why you were going around with smiles on your face. First of all, we asked you take stuff out, not to put it in.<laughter>

**Cline:** Oh, absolutely. <laughter>

**Cavlan:** Number two, you couldn't care less the part didn't work; nobody had to ship it.

**Cline:** Yeah. What a waste, it worked. <laughter>

**Cavlan:** Yeah. And, you know, you'd be a big hero for not doing very much. <laughter>

**Cline:** Yeah. But that's really the business story of programmable logic is that customers, they have to be able to see that they aren't going to have to pay an arm and a leg forever. They've got to have a plan for cost reduction but, because they rarely do that because of opportunity cost in their own engineering force, they end up buying programmable logic longer and that's really the story of the success of programmable logic.

**Cavlan:** You give an excuse for a design engineer to go to his boss and say, "Let me put this part in because I can always cost reduce it later, okay?"

**Cline:** Yeah, that's right.

  

**Cavlan:** And so the guy said "okay", but didn't really know what is involved. And, later on, the same designer was put on another project and all was forgotten.

**Cline:** Yeah.

**Haines:** So the product was out so now, this is 1975, so, Napoleone, then you launched the product. Can you tell us about the launch process, how you did it and what the outcomes were?

**Cavlan:** Yeah. Well, I remember the process very well because Ron and I had to put our skulls together to come up with a story that would introduce the product, but also would extol its benefits and put them in a light that was better than our competitor's. You know, Intersil introduced a device earlier and there was this contest in the marketplace going on. We got together, we talked about it and we went in front of a magazine editor and we had an interview, <turning to Cline> okay? And then, out of the interview, the editors issued what they called a position paper and that paper was published. I have a copy right here. It was in the Electronic News of July 21, 1975. It says, "Before you proceed into FPLA technology, consider this required reading." Wow, it looked like, almost, you know, something like criminals appearing in your neighborhood. <laughter>

**Cline:** I don't think there was ever an ad before that had that many words on it.

**Cavlan:** Right. And now this was considered an advertisement. I mean, if you will take a look at it right now, people in advertising would cringe, but anyway this is what we did.

**Cline:** You've got to understand, at that time, Electronic News was a tabloid newspaper so this was put on a sheet that was about yea big.

**Cavlan:** Right. Yeah, so this is a shrunk version and our names are on it. Ron Cline and my name are right on the top. It says, "What is an FPLA?" and so forth. But this was the first introduction of our product. Then this...

**Haines:** And you had to describe what an FPLA was because not very many people knew?

**Cavlan:** Well, I had to describe what it was because, first of all, FPLA was the first time the name occurred. It meant Field Programmable Logic Array. But also because of its internal structure in the applications we were trying to address, I had to explain to a potential user that he could look at it in terms of its architecture as a logic element or a sparsely populated PROM – and this was the dichotomy that existed at the time – to try to penetrate the mindset of the engineer. So that's what this tried to do, this paper. Eventually, this paper got attached to a magazine article in EDN, which was published on September 1, 1975 and I have the same page produced here and this article is printed in the back. (The video-camera can't see that very well.) At the same time, there was a press release that I'll talk about later, but this is the thing that I wanted to say regarding that article. This is a copy of the article that I submitted to George Rotsky. It was a collaboration between the two of us; <turning to Cline>. We wrote

  

this article. George Rotsky of ED Magazine took half of this article, essentially the one that talked about the product architecture and programming, and published it in his magazine. The article is reflected in this little brochure that we made. Then the second part, which is more centered on applications, was published a year later in the April issue of the 1976 EDN Magazine, which I have here. But the story about this paper is the fact that I was under the gun to give it to Electronic Design, to George Rotsky, and so I took the draft home and I went over and over it, making corrections and even though I think it was George Conner's wife, who was at the time the secretary...

**Cline:** Julie?

**Cavlan:** Julie typed it, but I had to make modifications. So, I took it home and I spent the entire night, both Nancy and I didn't sleep – my wife, Nancy – we didn't sleep. She typed. I made corrections. We cut and pasted this thing, this document and then delivered it. It was a labor of sweat as well as a labor of love, but we made it. So, that was the actual product introduction. Then the other part of it, <laughs> which is another interesting tidbit, I have it here, is the picture of the integrated circuit package that we gave to Electronic Design to use for the press release, which occurred in July, the same issue, the same time frame. The funny thing with this is that we, at that time, did not have first silicon yet. So what I decided to do is to take a ceramic package that was empty, no die in it, label it with the name of the product…

**Cline:** Statute of limitations. I think we're past that, right? <laughter>

**Cavlan:** And gave it to them to publish. So that's the way it was.

**Haines:** So you were confident they weren't going to unseal that lid and look at it?

**Cavlan:** No, they would not have the tools for that, or the inclination. But later on… Of course, I didn't do this just underhanded. I mentioned what I was doing to my boss at the time, Ralph Kaplan. He was the marketing manager of the department, not of the Division, of the department. He just started to call me "snake" from that point on. <laughter> But it worked, so it was all right.

**Haines:** So you got a lot of press around the world. You had to educate people. What else did you have to do to educate people?...

**Cavlan:** Well, following that introduction, I had the work cut out for me because, essentially, it came into a market that was completely untrained and unexposed to any of these things. There were only a few customers that were familiar with this type of product and few ever used it, especially because it was not yet field programmable at the time. I had to go out there and organize several seminars at different customer sites as well as public seminars sponsored by our distributors and organizations like Wescon, Electronic Engineering Times magazines, to go out there and explain to them what this device was and how it could be used and how it could be programmed. So, I made so many trips around the country and trips around the world. Eventually, I had spread the word everywhere and I was doing this single-handed. We did not have an organization internally that was tailored to be multi-pronged to address all these

  

things, so it was a slow process, a laborious task but I carried it, more or less, pretty well because, eventually, the word got around and, in fact, I was going to show you a measure of success that we had at the time. We knew we had arrived. We knew <laughter> that we finally broke in the market, they knew what an FPLA was because, at that time, there was a weekly magazine, a paper, we used to call it...

**Cline:** A newsletter.

**Cavlan:** ...a newsletter. We used to call it a rag. The author was Don Hoefler, and he used to...

**Haines:** We think it's the fellow that invented the name “Silicon Valley”, right?

**Cavlan:** Yes, yes. One issue you see here was around the beginning of maybe 1976 because of some dating involved on this document, on this page. He mentioned about finding a kind of graffiti in the toilet in the Sandpiper Restaurant in Cupertino at the time, referring to the FPLA with an expletive. So I don't want to repeat that, <laughs> but we knew that we had made progress there.

**Haines:** So you made Don Hoefler’s magazine, rag, so that you knew you'd made it. So you went around the world, you introduced this. What did customers start doing with the product? And who were the customers?

**Cavlan:** Well, the way we addressed the product, as I mentioned before, was to try to target applications that were byte oriented and bit oriented. So the first area where we tried to educate them was in the replacement of random logic glue that was used on a PC board to hook the system together. Basically, what you would be doing is lifting the traces off the board, putting them on a chip where they could be manipulated painlessly. A lot of benefits are derived from that. In particular, one of them is that you could reduce the number of packages required and the board area, but also you would be able to modify your design by avoiding PC board changes. And, in so doing, you also would hide your mistakes from your boss. Also, you could customize the product in the field. So the idea of using that device for random logic applications was very well motivated and people started using it as memory to microprocessor interfaces and microprocessor bus interfaces, simple decoders, I/O port decoders, those kind of random logic groups. There were also other areas of application in which the customer was prompted to use FPLAs: developing more custom functions that were a little bit more organized. These were available in TTL packages, but were fixed. For example: multiplexers, arbiters, decoders, encoder/decoders and so forth. Those were more like random logic but more organized. I like to put a division there because there is another area that I like to talk about which, in my mind, had much more organized, structured forms, similar to a logic table that looks like the address-data pairings in a memory, where you put in a particular binary address and you get a binary output – a lookup table. Those functions you would find, for example, if you built a digital combination lock or a frequency table for frequency synthesizer. Other applications that we were really targeting, and that customers found very good use for, was in building bit slices, microprogrammed bit slices, in which you have to develop branch logic. Also, in microinstruction decoders where you had variable formats it was ideal to use an FPLA because you could use ”don’t-care” inputs in the bit field coming in.

**Haines:** So these would have been the minicomputer manufacturers, the mainframes?

  

**Cavlan:** Minicomputers but also microcontrollers. In those days, you'd be using bipolar bit slices for all kinds of microcontrollers, industrial controller applications, and so you needed a microprogrammed machine to do that. That's how you would-- in fact, Signetics had the 8X300 product line that was oriented to that, and Intel had the 4004 and the 8008, and the 6800 for Motorola. Those are the kind of application environments where these products fit. Then there was another application area that was even more structured. This was in the area of memory overlays, where you would be using a small memory with a particular code stored in it that would be substituted for certain other areas of main memory that needed to be modified for many reasons. One might be for diagnostic testing purposes of the system, another might be to customize a product in the field that certain manufacturers sent out – machines that had ROMs in it. But then they added field programmable memory, and the field programmable memory was basically a few FPLAs that would contain the code that would be substituted… but I guess we can continue on later.

<break>

**Haines:** Okay, Napoleone, so you were talking about the different kinds of applications. You just talked about ROM patches.

**Cavlan:** Right. I was mentioning an application, the area of memory overlays. The interesting part was to be able to use the FPLA to essentially overlay different memory programs into another area of memory over a very large address space. The FPLA could manipulate 64,000 address locations because of its 16 inputs. So one could take a small memory and sort of implant it into any particular segment that one needed for that particular purpose. So, the application of the FPLA was very useful in those areas, either by storing the code directly into the FPLA or, if you wanted to do an overlay on a byte basis or needed to do it on a segment basis and there wasn't enough room on the FPLA, then you would use the FPLA as an address translator, as a pointer to another small memory, (could be a ROM or more effectively a RAM), which could store whatever code you needed and then insert it at the appropriate location in the large, main memory space of the program. And that would not only serve as diagnostics but also could be used to customize the mission of the equipment in the field. So, a customer could make one basic product and then sell it to different users for meeting their own particular specifications. But, beyond that, we did a lot of work in trying to address the market. For example… talking about conferences and seminars… this <showing proceedings> was the Electronic Engineering Times, Integrated Circuits Applications Conference, which I participated in. There are FPLA articles in here but what is critical here is the time frame. This was in 1976, yes. It's a 1976 document, so you can see the early activities that were going on to diffuse this device in the marketplace. Then, the worldwide effort that I undertook to disseminate the FPLA worldwide… this was one notice by Mullard, which was a firm that was essentially a sales-- well, more than a sales firm, an organization that belonged to Philips. They...

**Cline:** In England.

**Cavlan:** They were based in England and they were selling all Philips products. When Philips acquired Signetics, they were selling our FPLAs there, so I had to go in that area of the world to present at seminars. Then there were presentations at Wescon. This is one typical paper <showing a Wescon pub.>. Then, eventually, somebody picked it up in Japan. This is a Japanese electronics magazine <showing a Japanese pub.> equivalent to EDN that has our FPLA. You can read in Japanese all the

  

FPLA news. Another one in France here by RTC Compelec, which was another organization belonging to Philips. There, by the way, my name kind of helped me a lot. I have a story to tell you later about what happened to me in England… but specifically in Paris, at a seminar. I got in town in Paris at about six o' clock in the morning after driving all night from Milan, a very adventurous traveling situation.

**Cline:** Milan to Paris.

**Cavlan:** Right, sort of. It was to be a very short hop but then the Italians decided to strike. To save money, our sales rep put us on a train and, you know, then the trains strike. We couldn't catch a plane because every plane to Paris was out; so we, in a roundabout way, [first flew to Brussels where the plane got grounded because of high winds. We then fishtailed on a bus to Paris, but we had to stop at night in the middle of nowhere because a pregnant lady got very sick. We had to wait for a radioed helicopter to pick her up, and for her bags to be dug up in the moonlight from the bowels of the bus. Later we had a long obligatory stop for the driver's meal brake. Finally,] we made it to Paris, and then I had to deliver my talk at eight o' clock… 8:30, this was the time frame. I hardly had any time to get myself cleaned up and go to give my talk. I wasn't really up to snuff, but the guy who introduced me realized the situation and said, "Well, let me introduce Napoleone Cavlan." When you say Napoleone in France… okay, all doors open.

**Cline:** He's Italian, you got to understand, he's Italian.

**Cavlan:** So, that kind of warmed up the meeting and people got a bit more forgiving. In fact, I said, "Please forgive me if I start slurring my speech too much because I've been on an adventurous journey here." But then, besides the customer education that I went through, I ran a logic contest in 1976. This was with Electronic Engineering Times, again. They ran this contest here in which I promised a few prizes. First prize was a stereo system with turntable, speakers, and…

**Cline:** I remember that.

**Cavlan:** You remember that? And, out of that, we had all kinds of people sending in their own uses and applications. And so as early as that time frame, 1976/'77, we were able to publish this application manual <showing a booklet> that has all kinds of applications in it. That shows you that the product was well received and our efforts were succeeding.

**Cline:** I don't know when it came out, so the history museum will certainly know, but the Commodore 64 had the FPLA in it. In fact, I recall, on websites that deconstructed the Commodore 64 so people could figure out the logic, it listed the components in the schematic and then, off to the side, you'll find a truth table for the FPLA so that people can emulate that. I don't know how it got into the Commodore 64 but there was another example.

**Cavlan:** Yeah. I'm not aware of that. It must have gone by me at the time, the Commodore development. I know that the FPLA found its way into many customers. There were hundreds of customers worldwide.

  

**Haines:** Right. So, in fact, it turned out to be the first commercially successful programmable logic family.

**Cavlan:** It was.

**Haines:** Is there some, you know, core thing you think made that happen or that you can point to or is it a number of things?

**Cavlan:** Well, it wasn't really one thing, in my mind. From my point of view, I can speak from the way the market reacted. The first thing that came to my mind was that the FPLA basically solved a problem. Once people recognized what it could do… it solved their problem, they started to adopt it. Their adoption also was contingent upon the fact that this product did meet their needs but also was available because, as it happened with some competitors, once you've designed it and then it's not available, they would be left in the lurch. So it was available. We had high fusing yields. Ron did a very good job in making the part that way, such that it was not very difficult to manufacture after we got over our initial humps. There was programming support that was available. We tried to address these needs with a low cost programmer and so, initially, the thing that people were struggling with was “how do we program it?” I don't believe our competitors had any programmers in the field. They probably did everything in-house. I put together this little box, it was just a bunch of switches, called the PR-100. We made it internally and we distributed it to key customers for free. Some others had to pay. That helped a lot. The other thing that also helped was that we would do free in-house programming so the user, once he had tested his code for volume programming, he could come to us and we'd do it for free. That was a key element.

**Cline:** You mentioned, used this phrase, it solved a problem.

**Cavlan:** Yeah.

**Cline:** Actually, from my experience, it solved problems we didn't even know existed and I think that's a key to really successful programmable logic parts is that they have a degree of flexibility and ease of use and understandability that allow customers or designers to say, "Oh, here's a tool that I can apply to something that we [Signetics] didn't even think of." Good programmable logic devices have that characteristic. I think, in my experience since then, products that are targeted at one-off kinds of applications and can do only that but do it very well are not as successful as the ones that are really flexible, really easy to use and allow the customers to figure out how to use it. Those end up being the winning products. I think, in programmable logic, that's always been true.

**Cavlan:** Right, right. The only thing that I'd like to interject into this perception is that as much as we, from the manufacturing point of view, as the supplier point of view, looked at these potential benefits, which Ron essentially is pointing out, this flexibility, not only to adapt the product but also to be able to quickly go through the design cycle and make it a forgiving device to use, so you weren't frozen in any way to a particular configuration which then you had to modify.

  

**Cline:** Oh, that was key. It allowed design engineers to change their logic at eleven o'clock at night and not tell their boss.

**Cavlan:** Right.

**Cline:** That was really key.

**Cavlan:** I think you wish you had an FPLA in your FPLA... <laughter>

**Cline:** I wish. Oh, we...

**Cavlan:** ...you could fix your programming errors…

**Cline:** That's been proposed.

**Cavlan:** ...design errors.

**Cline:** That's actually been proposed but we couldn't do it.

**Cavlan:** But the thing is that, if you look at the way the story emerged in the marketplace, our competitors at the time that came later, well, each one tried to diminish the advantages of the other – point to the flaws. They were singing a different tune. They were saying that our parts were difficult to use and difficult to program. Well, we solved a lot of problems but didn't solve all of them, of course. We had a lot of benefits but not all of them, and the perception that they were difficult to program, the way it manifested itself (they said) it was because we had the two arrays, and so it was difficult for the engineer to understand the part and to program it. But that was a fallacy. If there was any kind of difficulty in the programming, it was the availability of programmers that were low cost and ready for them right at the time when these products came out. Then there were some legitimate complaints, the fact that the package was wider than what they really wanted. So there is this kind of dichotomy that emerged, Ron, and I think it's worth to point out that there were a lot of benefits but also there were some issues.

**Cline:** I think the industry learned that, for every dollar you throw into design, whether it be silicon design or whatever, you better throw a lot more dollars at the customer software, the development system, the support, everything that goes along with that.

**Cavlan:** Right.

**Cline:** And we didn't have that. We had you.

  

**Haines:** So that's a good segue into the development. Before we transition to that, though, this was the first very successful family. I think, Napoleone, you had a story about, in fact, these were so popular devices that some were stolen from you at one point.

**Cavlan:** Well… I think I need to correct the perception here. These were not the devices that were stolen. What was stolen from me was a set of PROMs. That was the time that Ron mentioned earlier when, after he designed the 4K in Schottky process, he then got the task, I think, to redesign the 1K PROM in Schottky.

**Haines:** So these are PROMs, not FPLAs. Okay. I misunderstood, sorry.

**Cavlan:** So, when I came to Signetics, my task was to fix what was a programming problem on the part. Because Ron had done his job, the parts worked well, he had good programming yields in-house on his system, but when shipped outside there were a lot of problems. Customer turned them back and there were concerns. They were raising ugly issues about the reliability of the nichrome fuse. So, what I did is… I took the new PROM that Ron had redesigned and looked at the actual fusing cycle. When the programmer applied the final critical step of putting a current into the device that should be clamped at a certain voltage, waiting for the fuse to blow, what really happened at that point? And the way you would observe that is to look at the output of the chip. That's where the two are coupled: a fusible link and a current source connected to it. A visibility port was one of the outputs and you looked at that to see what the waveform was. What I discovered was that the programming card that DATA I/O had for these PROMs was a voltage source that was used as a current source. One of these standard 3-terminal voltage regulators I/C devices was used in that manner, but they didn't put on a voltage clamp, so the voltage was free to rise to whatever level given by the value of the current times the resistance of the fuse at the beginning of “blow”, and the voltage went well over about… I think it was 19 volts that you <to Cline> had?

**Cline:** 21 volts.

**Cavlan:** 21 volts, it was way above that and what it would do is crush the collector to base junction of the...

**Cline:** I'm sorry, you're right. 19-- 21 volts was the limit so 17 volts was the spec and so you're saying it went above that.

**Cavlan:** Was way above that, okay? And so it just destroyed the output stage. It was not really the fuse that was the issue. So, while I was observing that and I was wondering why it went so high, I just grabbed a 1K resistor off the bench and shoved it across the output of the PROM and, all of a sudden, the problem went away. It was a big hurrah! So then I had to embark on a trip around the country to show that, indeed, our 1K PROM was trouble free and it would function with no problem. I took a bunch of parts and a programmer to do demos. I was visiting a customer in New Jersey. The sales people in the area I was visiting with… we had dinner and drinks in a bar [and most got “well oiled”. I was the only sober one because I never touched alcohol. They also had arranged for a follow-on visit with some ladies of the night, but I refused to go and waited for them in the bar. After they came back, the drinking

  

continued and one of our guys got rowdy and mistakenly crashed a removable section of the bar counter to the floor. The bouncer pounced on him and we all had to get in the way to prevent further mayhem. We were thrown out,] and then we left [for our motel at 3 AM.] In the meantime, I had visited another customer earlier, it was a military customer who had some PROMs that had failed, some parts that had failed, so he wanted me to take a look inside the package and see, and issue a report of what was the failure mode of these devices. So, I had taken these failures and put them on the same foam pad where I had the new units that I brought with me for a demonstration to customers. When I went to New Jersey to visit the customer and do my presentation, I took my programmer and reached for my parts, but there were no parts – the entire pad had disappeared. I was extremely embarrassed because I was new at the time and my Division marketing manager was there, and he had gone through all this spiel on how we had solved this problem and looked like a fool with no parts. So, we just left and eventually we recovered by some other means. But I learned, a few years later, that what had happened is that somebody during that restaurant gathering had gone into my briefcase and, knowing that we had these good parts, stole them and in so doing had stolen also the parts that I was supposed to analyze, causing me a lot of grief. Then, at that same time, I had these parts showing up on my desk. Somebody from Connecticut sent them to me with some kind of a story: “So we found these parts laying on a desk. Maybe they belonged to the bipolar memory Division, so here they are.” That was a strange event. It was like my Christening in the field. <laughter>

**Cline:** I'll say.

**Haines:** That's an interesting Christening. So one more topic before we move to the development tools, which was second sourcing. Second sourcing was a very popular phenomenon at this period.

**Cavlan:** Right. Second sourcing. Yeah. Well, the second sourcing, Ron, you know about...

**Cline:** Well, I know about the one, that was a big deal to me.

**Cavlan:** Yeah. There was Intersil that came first. Well, I'm sorry, Intersil, it was not a second source… I'm sorry. Was MMI, I'm sorry. Intersil was a competitive device… it was Monolithic Memories...

**Cline:** Well, that's a big one.

**Cavlan:** Yeah. Go ahead.

**Cline:** Well, they were our competitor.

**Cavlan:** Right. They were one of our competitors.

**Cline:** And this was a mask second source so the idea was we were going to give them all of our masks, give them our schematics, Scotch taped, I'm sure, and, to me, that was-- I mean, I was naïve but it was like my high school football coach saying that he was giving our playbook to our big rival. So, yeah, MMI

  

was our chief competitor in PROMs, had been for years. The story was, it ended up being that they couldn't manufacture and for a very strange reason. Not that they didn't have good manufacturing, it was very good, but the Nichrome fuse that they used, remember I talked about the challenges that we had with our nichrome fuse of putting enough current in it to blow?

**Cavlan:** Right.

**Cline:** They solved that problem by having a higher resistance fuse that took less power to blow---they took more area. They only had a single layer of metal process. They didn't have the current capability to deliver it in their PROMs so they developed this higher resistance lower power fuse that took a lot less current to blow but it took more voltage, v squared over r, for the same. So we had 100 ohm fuse, they had a 400 ohm fuse, which they tried to put into our mask set and-- because it required more voltage and because we had that emitter array, now the emitter array had a base emitter breakdown, avalanche breakdown, we called it a Zener breakdown but avalanche of about 6.2 volts. We were able to get the four volts across that nichrome fuse but they were not able to get the five and a half volts that they required by the time that you added all the parasitics. So they simply couldn't yield our product. It was a process mismatch.

**Cavlan:** Right. But if you look at that whole incident that John… Ron just… well, I'll tell you why, because I just made a connection in my head.

**Cline:** I know who you're talking about.

**Cavlan:** Yeah. It was a blessing in disguise for MMI because they managed very successfully to turn that failure into enormous success.

**Cline:** Yeah.

**Cavlan:** That is really the fundamental issue underlying the future development of these products, you know…that you take that a crisis and you make an opportunity out of it; and they succeeded in doing that.

**Cline:** They couldn't program an OR array, so let's not program in our array, let's make [a PAL]...

**Cavlan:** That's right. Then of the other potential second sources that emerged, that really were real second source plans, one was from Fairchild. In fact, Fairchild copied our device verbatim and I was lucky to find a datasheet… right here you can see it. At the time, they were trying to make this part, and actually they did make it, with… using the ISO planer process. It was Dr. Peltzer...

**Cline:** Dave Pelzer.

  

**Cavlan:** Dave Peltzer's baby. But it was a difficult process and didn't yield very well, so they finally abandoned the project. The other one, the other company, Ron, was AMD. They announced the future introduction of an FPLA just like ours, you know – compatible – in 1976 but, as far as I know, it never happened. I never saw one of those.

**Haines:** So, a few minutes ago, we touched on the development tools as one of the factors. Ron, you made the comment that this established the need to not only invest in development but to invest in tools and programming and customer education. So let's talk about the development tools. What was available to the customer and how did they use these devices?

**Cavlan:** Well, you know, in the 1975 through about 1981 time frame, there were no personal computers available. IBM PC didn't exist nor was Apple there. So, it was difficult to conceive of developing any kind of software to put in the hands of an engineer to do designs in his lab. That was not even on the table as far as we were concerned because it was a different story. There were PROM programmers, of course, and they used a program table that the design engineer would generate, basically by hand or maybe by some kind of a little Mickey Mouse in-house system that had automatic generation of paper tape or a bunch of punch cards. But that was the design style; those were the design tools of the day. They had access to mainframes, the designers, for certain tasks but that wasn't the right environment to do any kind of logic design and development from the customer point of view. We had a device that basically, if you looked at it, it was a sparsely populated read-only memory. It had a large address base, 16 inputs and an output field of 8-bits. But, in terms of the depth of this memory, it was very shallow but it looked like a memory. So we said, okay, how do we tell somebody how to define this logic function? Well, let's give him a program table that looks like a memory table, and let him put in symbols that stand for the response that he gets at the output and the voltage levels that he programs in the inputs for these variables, logic variable inputs; that will tell him what the function of the device will be.

We had to come up with a special set of symbols to do that. I mean, H for high, L for low (zero, ones) and then, in the output area, we had to designate whether an AND gate would be activating an output, so we chose the symbol A. And then there was a big deal about what do we do when we don't want an AND gate connected to an output. How do we designate that? I decided to have a “dot (**.**)” for that. That caused a little bit of an issue between me and DATA I/O, the programmer manufacturer, because they felt that it was very easy to use a “dash (–)” to denote the fact there is no connection, and also use the same (–) in the input field to program the condition “don’t care”, where both the true and complement values of an input are not going to be programmed into an AND gate: in other words, totally disconnected. Now, I wasn't happy with using just three symbols, H, L, and (–) for defining the program table because, in my mind, I knew that eventually we'd be developing some internal logic paths that we also had to define, and we might require a different symbol for that. That turned out to be prophetic in the FPLS, another device we did later. So, I remember having many debates with DATA I/O on the phone and, eventually, we committed to buying one of their Model 10 programmers, the first one that came out – it was $8,000. So, I went to visit them. They had assured me that, "Yeah, we're going to do this. We're going to implement your symbols." Then, I went to look at the machine and it didn't have that. They only had the keys H, L, A and (–). There was no (**.**). I wanted a (**.**). So, there was one key that was labeled (–), and I said, "You guys need to change that to also allow keying in a dot (.)." And they said, "No way we are going to change that. We have built all these machines." (The real thing is that they ended up with only a few.) I said, "Well, you have to do that, otherwise, I'm not taking the machine." Wow, that was a big row but, anyway, they eventually were gentlemen about that. They agreed, and not only had to make software changes inside to recognize this new character, which they could easily do, but also they

  

had to change the button on the machine to include both symbols on it. But it was resolved, yeah <laughs>. So that was the “development tools.” It was basically a truth table, program table chart in a programmer that enabled to do that, program the logic function. There was no simulation tool. There were not Boolean expression tools. You could not write a logic equation and eventually embed it in there. There was none of that. In retrospect, that was not a very good way to prioritize development of our support tools but that's the way it went. We did have, later on, in 1976, another aid that we generated for the users. That was a logic minimization program. <turning to Cline> Do you remember that?

**Cline:** I...

**Cavlan:** LogMin. Did you...

**Cline:** I didn't take part in that.

**Cavlan:** You didn't take — yeah. That was a logic minimization program we developed. We financed it but we had development done at a university. This logic minimization was a piece of software that ran on our in-house computer and was accessed via remote terminal by a customer. He would input his code at the terminal and then we would process it and send him back the result. The idea was that he submitted a program table, which was a certain size, it had so many AND gates or product terms, and then we would come out with a minimized form so he could fit this logic table in the FPLA. Let's see. The program, as I mentioned, was developed at Stanford University. The person who was involved in actually writing code was a student of Dr. Ed McCluskey and, as I said, we financed the project. It took about a year, a year and a half, to develop. LogMin, which was the name of the program, was an extension of the...

**Cline:** Quine-McCluskey.

**Cavlan:** Quine-McCluskey method that most logic designers who took a design course would be exposed to, even nowadays. But that was an attempt to extend that algorithm to manipulate eight outputs, each with 16 input variables. There was simultaneous minimization. That was a very difficult thing to do. Sometimes it succeeded and sometimes it didn't. When it didn't, I appended a note at the end saying to the user, "Congratulations, your program table is already minimized to an optimum number of product terms" and he would not know that what came out of the program was a larger number of product terms than what he had put in... <laughs>

**Cline:** One thing to set the context of all this just for people to realize is there was no PC at that time.

**Cavlan:** Right.

**Cline:** Nobody had any tools.

**Cavlan:** Yeah, nobody had those things.

  

**Haines:** Right. Mostly it was customers probably using the truth table method; they'd just do it by hand and they'd put that truth table, you know, program it into the programmer, the data I/O programmer, and that was the mechanism for principally for doing it.

**Cavlan:** Right.

**Haines:** Okay. How many fuses had to be programmed or blown for this and how long did it take?

**Cavlan:** For the FPLA, it was about 1,500.

**Cline:** Roughly 1,500.

**Cavlan:** 1,500 yeah.

**Haines:** That was the total number of fuses or the average number?

**Cline:** Total.

**Haines:** It was the average...

**Cline:** We first did 1K and 4K. It was in between.

**Haines:** In between. Okay. So you'd program some number, presumably some percentage of those got programmed every time?

**Cline:** Yeah, but typically more than half because, in order to have those "don't cares" that he was talking about in the AND array, especially, you had to blow both fuses that were associated with that input on that P term. So we ended up blowing typically 80% of the fuses on any given design.

**Haines:** And about how long did that take in the data I/O programmer?

**Cavlan:** Well, okay, Ron, the program pulse, I think, was about...

**Cline:** It was spec-ed at one millisecond.

**Cavlan:** Yeah. It was about that long. Then there was a cycle that the programmer had to do. You had to do...

  

**Cline:** I remember watching the model 10 so it took between five and ten seconds, I think, to do that.

**Cavlan:** Yeah, it was about that much, ten seconds, because of the overhead in addressing...

**Cline:** That's right.

**Cavlan:** And this only — if they did an initial verify to check all fuses initially intact, then they did a program, followed by another verify cycle.

**Cline:** Another verify, yeah.

**Cavlan:** But then, the Data I/O Model 10 also had a “logic” verify algorithm. I pushed them in that direction. In fact, I even published, in one of my papers… on our paper together, <to Cline> remember, there was a section in which I put a block diagram on how to structure...

**Cline:** So would you derive the logic equations out of the pattern and then exercise it.

**Cavlan:** Right. Right. They would exercise it. And so, when that happened, it took about a minute per device, I think that's about...

**Cline:** Well, 16 inputs, that's, what, 64 K possibility.

**Cavlan:** Right.

**Cline:** Times eight-- yeah.

**Cavlan:** Right. And so, that was about the cycle for each part.

**Haines:** So it was very fast, basically.

**Cavlan:** Well, it was fast but some people wanted always...

**Cline:** It's never fast enough.

**Cavlan:** ...faster than that… yeah, never. And so the effort of a lot of customers was to squeeze us into reducing the programming times and the setup times that were required, that we put in our specs. Then there was one other issue surrounding that. We could not make the programming algorithm proceed too fast because the part would heat up. If you essentially...

  

**Cline:** Oh, I forgot about that.

**Cavlan:** Yeah.

**Cline:** We had to put in a wait cycle.

**Cavlan:** Yeah, we had to put a wait-state, because otherwise you would say, okay, you want to go fast… put a piece of ice on it! But that couldn't work in practice. So we had to do this the wait-state, but I think the problem was addressed later by the programmer manufacturer in two ways. One was to cut the overhead in addressing and manipulating the part. The other was to create gang modules that would program several parts at the same time. So that was a way to mitigate the length of...

[Eventually, programmer manufacturers discarded the "logic" verify feature to reduce the cost of their systems and the total programming time per device. This was based on the accumulation of field data showing that our FPLA was immune to P-Term conflicts, such that once its programmed fuse pattern was verified, "logic" verify never failed.]

**Haines:** Right. So you had your own programmer. DATA I/O was one of the dominant manufacturers; they supported you. Were there other guys that had programmers?

**Cavlan:** Yes, there were others that came later. In fact, I have to mention that the DATA I/O model 10, even though it was the first machine and very expensive, played a critical role in ensuring the success of our FPLA. But there was another manufacturer, Jack Curtis. He was an ex-Signetics employee who had left the company. <to Cline> Remember Jack?

**Cline:** Yeah.

**Cavlan:** He created a small programmer just for the FPLA and I helped him build one with the specification of what the machine should do, and in developing the algorithm. So, his was a very critical role in that success. And then, of course, there was my programmer, the smaller one. But, beyond those, there were others that emerged, especially in Europe and also back East. I think in Florida, there was one. One of them was Sunrise Electronics that was a manufacturer of FPLA programmers. The other one was Kontron, based in Germany. I remember working with those guys very well. One guy came to see me all the time and traveled to Europe, yet they were based here but the engineering was done in Munich, I think. One day, he came...

**Cline:** You should understand that every time he mentions a programmer, that it had to be qualified and that meant him working with them to make sure that he wasn't going to have a non-voltage clamped output any more. <laughter> The Signetics' reputation for reliability depended on those programmers doing exactly what they were supposed to do.

**Cavlan:** Right, right.

  

**Cline:** So it was a big job.

**Cavlan:** Yeah. Well, that was an additional part of my job. Not only the application and product support and being an evangelist for it, but also taking care of all the manufacturing guys.

**Cline:** So, it's important to understand. It's one thing to say, yeah, there was one designer but it was a much simpler, less complex part than what we talk about today but there were all these other functions that have completely separate groups nowadays in modern programmable logic companies, they were all wrapped up into one person. [Cavian,] when you're doing things for the first time, it's not recognized that this particular task, when I say recognized, I mean by the company, in terms of value, it's not recognized that this particular task does require a lot of resource and effort.

**Cavlan:** Yeah. We were essentially in a bootstrap mode in those days, just trying to find ways to develop product support, and so Kontron was one of the key ones. In fact, I was saying I remember the guy because he went on a trip to Europe and asked me what I wanted back from there. I couldn't think of anything and then I said one day, "Bring me some truffles." So he brought me the most delicious Belgian truffles I've had in my life.

**Cline:** I never got truffles. I never got anything.

**Cavlan:** Yeah, I know, I did. <laughs> And then, of course, there was another one, you know, that was also a very good developer for us in England. It was Stag. They had a set of programmers for PROMs and for FPLAs, low cost, production oriented. And so they had a spectrum of machines, the same way DATA I/O had here. Kontron was small at the beginning; eventually, they grew. But these are the guys that helped us in having the product succeed in the marketplace.

**Haines:** Okay. I think it's time to move on to the next topic.

**Haines:** So the FPLA was a successful device and then you moved the technology into other products. You expanded the product family into what became called FPLF, the Field Programmable Logic Family.

**Cavlan:** Right.

**Haines:** Can you describe what was the idea, how did that come about?

**Cavlan:** Yeah. Well at the time that we had the FPLA out, and I'm talking about again in the 1976-79 time frame, the feedback that I heard from the field really was really basically two things. They wanted a lower cost part.

**Cline:** Of course.

  

**Cavlan:** And the other thing that they wanted to do is… they were having difficulties using PROMs and registers in the output. So, they wanted to have PROMs with flip-flops on-chip so that they could implement state machines. Therefore, the natural extension for us was into an area of products where somehow we could satisfy these two requirements. And how did we do that? In my mind, the first thing that we did… well, actually there was one more, but I think you <turning to Cline> probably had the larger role in that one. That was what I called the ROM patch… and we'll talk about that. But the main things were extending our product line with a lower cost device, and then something that could be used in place of PROMs and registers. So what we did then was to take the FPLA basic structure. You <turning to Cline> had formed a design base with the FPLA, solved the inherent design problems, and you again were asked to throw something out, which you loved to do for getting a small die size. So you took the OR array out, and just retained the AND array, just the AND; and instead of 48 gates, there were only nine gates left, nine gates, right, because there were nine outputs. The input field remained the same, 16 and the output was nine, so that's where the nine —

**Cline:** A smaller chip to make it faster. I loved it.

**Cavlan:** Right, so there was just nine AND gates with programmable output polarity, with which you could do AND or NAND functions. So, that basically meant scaling down the product considerably, not only from a architecture point of view but also using only about 300 fuses, and it was faster. It was about 35 nanoseconds, and much lower cost, less than $1.

**Cline:** That's right, yeah.

**Cavlan:** So we did that. Then the other thing that came up—

**Haines:** And, Napoleone what did you call that device?

**Cavlan:** Well, all right, I'll tell you now what it's called. That's a very key device in the history of programmable logic. Because the device had a single logic level on it, a single AND, I decided to call it a field programmable gate array. It was just a flow through architecture of simple gates. What is ironic is the fact that now the name of the simplest device in the chain of field programmable logic architecture today designates the most complex devices at the other end of the spectrum of programmable logic. But at that time that's what I did. I took the FPLA apart and made a simpler and faster set of wide decoders out of it. That's the origin of the name "field programmable gate array".

**Cline:** Just a question of scale, right?

**Cavlan:** Right. So, FPGA… that's when it was born, the 82S103 (or PLS103)… That was the simple part, and then we had the next offshoot, the FPRP (82S107), which was the field programmable ROM patch. I believe you <to Cline> were more instrumental in getting some input from some customers. They wanted a way of patching memory, memories like large ROMs or even core planes where these core planes were delivered with defects. To improve the yield that they were getting from the manufacturers they would put an FPLA in parallel with it and then detect the failed address and jam on

  

the output bus the correction stored in the FPLA. But in order to do that directly, they needed a flag to select which data source to put on the data bus, so you added one more OR array gate hardwired on the —

**Cline:** And just went high anytime any p-term went high.

**Cavlan:** And that's the way it was designed, so that was another part added to the family. And then in conjunction with that, while you <to Cline> were doing all of this, you…the quick draw designer, designed not only all these parts but also another one, the FPLS (82S105, PLS105), which was the field programmable logic sequencer. And it was named that way because that part allowed the user to do self contained state machines. It had an internal state register as well as an output register. It basically did the function of a Mealy state machine in which the outputs are a function not only of the present-state but also of the input. So, your machine could basically idle in the same state without consuming internal P-Terms while receiving many inputs and changing the output stream to whatever other parts needed to be controlled that way in the system.

**Cline:** We had a lot of conversations about that one, I remember, because that was a much more complex part with having eight programmable registers in the output.

**Cavlan:** Right, right, right. I wanted to put more flexible registers on the chip.

**Cline:** Of course you did.

**Cavlan:** Because I always asked for more, but Ron couldn't do that because it wouldn't fit. I had to compromise and said, "Okay, then let's do just Set/Reset Flip-Flops." I didn't think that a D-type Flip-Flop, which would require only a single OR-array gate per Flip-Flop, was going to be the best for a —

**Cline:** You couldn't do a state machine. I agree with you on that.

**Cavlan:** You can make state machines using D-type Flip-Flops but not very efficiently in general because the D-type Flip-Flop just by itself requires its state to be re-asserted anytime you don't want to jump state. To keep it idle you have to use another product term. Now, you could use a D-type Flip-Flop, feed it back into its input, and use another gate on the IC [Integrated Circuit] package to convert that into something more like a Toggle Flip-Flop or a Set/Reset Flip-Flop – but that is essentially using product terms inside a part in a way that is not efficient. For state machine design really the best is to have Set/Reset Flip-Flops. But now if you have a J/K Flip-Flop, then you can use it as a Toggle or D-type Flip-Flop, and then you also have the ability to best match its function to the application.

**Cline:** I couldn't give you that.

**Cavlan:** Right.

  

**Cline:** Primarily power considerations because in order to meet the performance at a certain frequency you wanted, the JK was just too complex to be able to do that.

**Cavlan:** Right. So the FPLA architecture was basically extended. It would retain the product terms, the spread of 48; we added the test array, and then we put another architecture innovation in there. This was in response to a customer request who had a lot of experience with state machines and anticipated that given the number of product terms we had on the chip it was easy for him to run into a situation where he would run out of product terms. And so he asked if we could generate complementary AND functions on chip so that when you made a state jump from one state to another you could minimize the number of AND gates required for that jump function by constructing a complementary version of the same function. And so I came up with a way to structure a Complement Array on-chip, and Ron looked at it and he came up with a very elegant version of that, which required adding only a single NOR gate on-chip. And the trick that I think was remarkable was the fact that a single NOR gate could serve to minimize all transitions in the state machine design because the inputs to all AND gates fed to the Complement Array were present-state dependent. Therefore, for each set of AND gates that were looking for an input to jump from a given present-state, once that state changed, all the associated product terms that were connected to the Complement Array were automatically disabled. So, now you were able to use the Complement Array again for another set of product terms associated with another present-state. It was a very elegant way of doing that. And so our state machine, the logic sequencer, became a very efficient and effective single-chip state machine in a logic design.

**Cline:** Can I mention one thing?

**Cavlan:** Yeah.

**Cline:** So the circuit design on that, I get to show something now.

**Cavlan:** Yeah, okay.

**Cline:** So many people here watching will recognize the front page of this. This is the standard form for the ISSCC conference, the International Solid State Circuit Conference. This was 1978 and at that conference I presented the circuit design for the FPLS and it was the first programmable logic paper to be presented at that conference.

**Cavlan:** Yeah.

**Cline:** It was a big red-letter day for me and for Signetics.

**Cavlan:** Yeah, that was very nice. So we then at one point in time— first of all, let me tell you that the first hints of what we were planning for this family appeared in this issue of the Electronic Times magazine. It was in 1976. The seeds of what we were going to do were already planted there. There's a picture of me in the paper here at that time <showing a magazine>, a much younger guy, and then inside

  

there's a description of what our plans were in a segment of the article. We were considering the next step in 28 pins. One of the things here that Ron also used earlier… they referred to me as Nap Cavlan, and Ron too called me Nap before…Nap. I didn't really like that name <laughter>. I am from Europe, from Italy, and we didn't use nicknames in my day. I don't know, maybe today we have become more global in customs, but in those days we didn't do that, and to me it was very strange. So, one day I went to Mike Hackworth, our Division marketing manager, and said, "You know, you better stop this and issue an official memo that from now on my name is Napoleone." And from that point that's the way it stuck. <laughter>

**Cline:** I didn't get the memo. I kept calling him that.

**Cavlan:** You didn't get the memo? Oh, you probably ignored it <laughter>. So this is the product line that eventually came into a booklet shown here in 1977 when, in advance, we published what these architectures were going to be. And then we showed this to the field within an advertisement <exhibiting a framed picture>, which was printed in the Electronic Engineering Times magazine. This, as you can see <pointing to the picture>, is a focus on the architecture of the FPLS sequencer that Ron designed, and then here outlined are all the four members of the family: from the simplest FPGA going to the FPLS, the most complex. It even tells you when we expected those parts, and what was available at that time. The FPGA at the time was available, so was the FPLA. The ROM patch was going to be available in mid-'79, and the FPLS in the fourth quarter of '79. I think we met that.

**Haines:** So you met those dates.

**Cavlan:** We met those dates, yeah.

**Haines:** So what was the relative success of these various family members in terms of adoption?

**Cavlan:** Yeah. In terms of adoption, if I look back I have to say that for the FPGA product set we initially were successful because it satisfied that segment of people who were looking for lower cost, but not those customers that were doing more advanced designs. So, we had moderate sales. Eventually, I believe after several years, the 28-pin version of that architecture was eliminated. The FPRP, which is the ROM patch, I don't believe we ever sold any of them. It was just a —

**Cline:** It was a perfect product for narrow applications. Remember what I was talking about before?

**Cavlan:** Right. Yeah it was just a security blanket for someone to know that they could use it if they needed to but, in fact, <turning to Cline> if I remember correctly, wasn't there one manufacturer that bought some and put on their shelves for inventory, and then they returned them to us?

**Cline:** It could very well be, really cool idea the way it worked.

  

**Cavlan:** Yeah. The FPLA was the best seller and the FPLS, the sequencer, was also a very successful seller but it was behind the FPLA in terms of total sales.

**Cline:** Well the FPLS was significantly more cost. It was about a 30 percent because of all those flip flops and the feedback state machines. It was a much bigger chip so it cost more too.

**Cavlan:** Right.

**Haines:** So this is a significant investment over a period of time, a year or a couple of years right? So was there anything different in the management sign off by this time? Was it more formal or were more people involved in this, or was it similar to the first round?

**Cline:** I was still a ground-level engineer. I didn't know.

**Cavlan:** No, it was essentially the same process going through the hierarchy. It was a development that originated in the engineering application area where Ron and I would be collaborating on these things. He would be talking to his design manager because he had to be clear about the feasibility of his parts and the die size to be achieved and the cost. This information would be discussed with marketing and eventually if marketing bought it, it was a very easy sell to the Division manager. He just went for it.

**Cline:** I think that there was a neat aspect to this period of time that there were no really other experts out there that could argue with Napoleone, for example. I think that probably never occurred again because we were right on the bleeding edge of defining products that had never existed before and capabilities that had never existed before. So I was the expert in implementing. I knew what we could do and not do and he was the expert in what would be interesting to do.

**Cavlan:** Yeah it was uncharted… navigating in uncharted waters, so it was very difficult to put in an opposite view and sustain it, as much as it was a difficult position for me to put a lot of faith in the product because all these products were just like tossing seeds on the ground and waiting for them to sprout, you see.

**Cline:** Yeah that's true.

**Cavlan:** It was flying by the seat of the pants, basically, if you really want to know the bottom line of that.

**Haines:** So you showed us some of the material used to launch this or you launched with advertising. Was your launch similar to the previous launch in terms of magnitude?

**Cavlan:** Yes, it was the same thing initially. We made an effort later to develop more sophisticated tools that did appear on the market, serving not only our 28-pin products but also the 20-pin family, which we developed later. The only thing that I can say is that we — well not the only thing, one of the things that I

  

can say is we were kind of late in developing those tools because of a lack of internal focus and in part because we kind of misjudged initially what were the best tools to address the market. We thought that manipulating Program Tables for devices of such complexity was not a very big task for an engineer. But what happened is that the engineer had to go and understand the architecture, kind of get it into his mind before he would program the device table. And some engineers were not willing to do that and so they were looking for other tools. I'm talking about design entry. Forget about the other issues of simulation and testing, vector test generation… needs that became apparent later. So the tools for subsequent products were essentially the same, but we did go into AMAZE later on which was a —

**Cline:** AMAZE, A-M-A-Z-E.

**Cavlan:** Let's see. I think I even have… yeah, you see <showing a brochure>… IFL software. At that time — the topic on the table here is the expansion of the FPLA into a 28-pin family, which we called FPLF, but eventually I changed that into a new name. I called it IFL, standing for Integrated Fuse Logic, which was basically motivated by our 20-pin family, and then I folded the 28-pin under it so that everything became IFL. And then when the software came out it was in support of all these products, and this is the description of that software. As well, <showing another brochure> this is a description of another device programmer that I also developed at Signetics in support of product introduction, which was another issue. So, in total response to what you said, yes, the 28-pin family entered the market with essentially the same level of support as the FPLA had, and we kind of really ran along with it until we were forced by user demand to do something different… which we did.

**Haines:** So you mentioned the IFL family. Eventually you started to look toward the next generation, the product expansion and some were more successful than others. Then you decided that it was time to make the transition to a smaller, cheaper package presumably.

**Cavlan:** Right. After a while, around the 1979-80 time frame, that's when the FPLS was introduced, it was clear to me in dealing with people outside, and even Ron had a lot of feedback on that… <turning to Cline> I remember you visiting Tektronix one time and coming back with some responses from them about our family of products. What was apparent was the emerging of a marketplace with kind of two areas, if you will, that you could designate. There was one area composed of unsophisticated customers that didn't really know how to use these parts and basically wanted to use them in applications where their cost was more of an issue and the I.C. logic replacement was not extensive. There were only a few packages to be replaced and they were looking for a smaller footprint.

**Cline:** A few gates.

**Cavlan:** A few gates. They were looking essentially for a less dense device. Then on the other side of it, especially as a result of the FPLS success, there was another set of customers. They wanted more flexibility and more features.

**Cline:** They wanted those JK flip flops.

  

**Cavlan:** Yeah, they wanted everything in there but they were just tossing random requests. It was not like somebody said, "Okay, why don't you do exactly this" or "Why didn't you put in that; how many of these?" or whatever. No, there was no target, no application that they had in mind for which we could aim a device to make it like a killer application for it, saying "Okay, if you do this we're going to use a million of them", no! They just said, "Oh, it will be nice to do that." So, there were isolated points that emerged from the different voices in the marketplace. I looked at all of this and then I had to take into account the competitive environment. And at that time, I think our PAL [Programmable Array Logic] competitors had already come out. Yeah, it was MMI. So what I did, I said, "Okay, rather than kill two birds with one stone let me see if I can kill three birds with one stone." What I said is, "Okay, I will come up with a new family of products in a lower cost 20-pin package, fit into a narrower footprint package, and will provide again the entire spectrum of logic functionality: single level logic like the FPGA, two level logic sum-of-products, the FPLA, and then sequential logic or FPLS." So these were the three basic structures. The FPGA I had to put it on the side as a unique part by itself because of its very low density. But for the FPLA and the FPLS I decided that I could combine them in only one device including an AND array, an OR array and J/K Flip-Flops that could be bypassed to the device outputs so that you could have a combinatorial function only. It was essentially an FPLA feeding on-board registers if you so wanted by not bypassing them; in addition, it had the Flip-Flops internally fed back into the AND array as well.

[Such arrangement would allow a user to implement self-contained sequential state machines of the Mealy or Moore type as desired, as well as a mix of flow-through combinatorial or registered functions. Most important, from a competitive point of view, my proposed architecture was a 20-pin PAL superset that could be programmed by the user to duplicate the function of 13 different PAL devices.]

When I presented this part to him <turning to Cline> he almost choked. He said, "There's no way I can put this together." He said, "Given our process limitations we can't do this." And so we argued about this for quite a while and then, finally, that one is one I lost.

**Haines:** Where were you in the process by that time?

**Cline:** Maybe one and a half micron. It had improved. It wasn't on the steep slope that it was in the late '80s and '90s so it was still pretty hard to put more transistors on there.

**Cavlan:** Yeah, and besides the package had shrunk and the number of product terms that I wanted to put in did not significantly decrease. It was a 42 product terms.

**Cline:** You wanted to put it in a 300 mil package which basically meant the die size had to be at least 100 mills, no more and that was tough.

**Cavlan:** Yeah, and the OR array would have to be doubled with respect to the FPLA. Then the J/K Flip-Flops also contributed to width, so it was really a tough nut to crack for Ron and he said, "This one I will not take the risk." I had to put my tail between my legs, go back and say, "Okay, now I have to split these parts." I had already put the FPGA (82S151, PLS151) aside, which at this time was an improved version of the first one. It had, I remember, 12 programmable AND gates that were grouped in three sets of four,

  

and all the outputs of these gates were fed back into the array so that you could dynamically or statically program their I/O's and use them as either input or output gates. Then the FPLA (82S153, PLS153) AND array was reduced to 32 product terms instead of 48, but then I added another 10 product terms so that I could dynamically or statically control the direction of the I/O's. That made it also more flexible than the original FPLA output polarity control. And then the FPLS (82S159, PLS159), that was the one that in a way was the most complicated device of all in those days. It was essentially an FPLA structure that fed eight edge triggered J/K Flip-Flops. The J/K Flip-Flops could be dynamically or statically converted via a control gate to function as D-type, as well as J/K Flip-Flops. So a user had the entire spectrum around him. He could use them as state registers in a state machine design or he could use them as a shift register requiring just one product term per Flip-Flop to shift a zero or one; there was no need to provide for both set and reset terms. They could also be used as Toggle Flip-Flops in building efficient multi-stage counters. I also put in a mechanism that would allow data from the output ports of the device, the outputs of all Flip-Flops, to be jammed directly into the input of the Flip-Flops so that you could dynamically turn an output into an input, bypassing the AND-OR arrays, load data from a bus into the Flip-Flops, and then execute some kind of logic function internally. This feature was also very useful for functional testing. I also added a Complement Array and various programmability functions for output enable, asynchronous reset/preset, and a clock line. It was a complex architecture that I then split in three separate configurations offering different output mixes of combinatorial/register functions. One had eight Flip-Flops (82S159), another had six (82S155, PLS155), and the last had four (82S155, PLS155). And so that was the structure of the family. In the IFL-20 total there were five parts… What was the issue with that? Yeah… well we're going into another area right now about the way it actually emerged. The FPLA of that family, <turning to Cline> Ron, was the first one.

**Cline:** That was the 82S153. {Corrected by Napoleone}

**Cavlan:** That was the 82S153 (or PLS153), which went on to be a very, very successful device. Ron didn't start the design then because he had left Signetics at the time. He wanted to get into another area of design and there was a better opportunity outside than within Signetics, if I remember correctly… Ron?

**Cline:** That's true.

**Cavlan:** So, we hired a new designer and I don't remember now if he was a consultant or what. He was put on the task but he had no experience in designing programmable logic and the design cycle was, yeah, it was long. And then he quit in the middle of the design so to speak, such that the part didn't get finished by him. So, what we did was to hire Ron back at the time in a consulting role, and you <turning to Cline> completed the design quickly after that. I don't think it took you too long to come up with it. It was 1981 I believe.

**Cline:** Nineteen eighty-one.

**Cavlan:** That the part came out.

**Cline:** I think I was just advising on it. I don't think I worked first engineer.

  

**Cavlan:** No, no you were not. You were not the first. You came in and basically completed the design. This was a side project for you because you were doing other things. And there was the time I remember when the first device came out, the first silicon…I discovered that he forgot to put in a circuit to verify the programmed logic polarity of the output. I was tearing my hair out. Now what's going to happen here? I remember you <to Cline> came in the lab and I told you about that and I said, "Boy this is ominous." And you said, "Don't use these kind of words." <laughter>

**Cline:** What I said---I don't remember that.

**Cavlan:** Yeah, I was very frustrated by that, so I asked him [Cline] to give me the circuit diagram and I took it home and overnight I studied it, tried to find out how the programming area worked, because he thought that he boxed himself in. He had no degrees of freedom anymore on the programming algorithm. I found a way where I think now it was adding a resistor and a couple of… sorry, a transistor, a couple of resistors and a couple of diodes at a point, and the breaking of a wire trace. And that was the solution, which made me very happy because he looked at it and said, "Oh, I can do this without changing the die size, or impacting the programming algorithm appreciably", so it was done. But that's when things… it was like the apex of Ron's involvement in this because after the success of the FPLA we marched on to the development of the FPLS. Again it was very difficult to find a designer, and the commitment of our Division was very slow to move on with this project. I got frustrated and then I too left Signetics. I left Signetics in the early summer of 1981 and went to Fairchild. I went to Fairchild to develop programmable logic devices there.

**Haines:** If I could interject here, so when you two gentlemen left Signetics essentially MMI took over the market not too long thereafter right?

**Cavlan:** Well it was not a takeover yet. They made very long strides but it was not a takeover. My leaving Signetics at that time was not the end of the story on that because as I left Signetics with the FPLS undone, I regretted that it didn't happen because of all these delays. But when I went to Fairchild, I was informed a few months later, it was around November of the same year, 1981, that we were slated to move to Puyallup in Washington. It was a long plan in place to do that, but I knew I would never leave California. I figured, "Wait a minute. If I'm going to engage in this kind of activity here and start a product line, I don't feel right that I stay now and then quit in the middle of it. I don't want to repeat the same situation with Signetics." So Mike Shields was at the time the process design manager at Signetics. He used to call me periodically to see how I was doing. I think he was trying to find a way to get me back, and so I did eventually agree to go back to Signetics, and at the end of 1981 I went back. Signetics then assigned another design engineer to design the FPLS, which became another problem again because this engineer was unfamiliar with designing programmable logic. He had done PROMs but not those parts. And so there followed a long, very long design cycle that was again hampered not only by his inexperience but his learning curve. Our tools, engineering design tools were still so primitive. The process had not significantly advanced from what we had in the last years and so the FPLS design cycle was very long and I think it was in 1983 or '84 that it was introduced. So you can see, I announced in August 1980 that we were going to be out with the entire product family by the end of '80, and it wasn't until about 1984 that we had all the parts. You can't play that game in the marketplace, and that is when MMI made its great strides over that time frame.

  

**Haines:** So together you went through several generations of product families at Signetics. You had made great strides with the first programmable, successful, commercially successful device. So when we transition here what lessons, what kind of final thoughts do you have about this? What kind of lessons learned?

**Cavlan:** Well, before we do that I want to finish the story because I… went back to Signetics, as I said. We finished the FPLS and it came out late. So, because of the fact that there were other competitive devices out there in the field that had a large measure of success, the natural thing for me to do was to say, "Well, what do I do next?" There were also other companies that were putting out much denser and complex devices. I think Altera at that time already had come up with something. So, in mid 1985 I decided to take a new approach and to design programmable logic in a different architecture that I called Programmable Macro Logic. PML essentially broke the AND-OR chain logic structure that we had used in the past and only used a NAND plane, a single NAND plane, which folded its outputs on itself to provide as many levels of logic necessary to optimize a function. The array fed, in principle, large fixed or locally programmable logic islands around it that could be defined in the future for any kind of a family of products that you would develop based on them, with different functions targeting different application areas. And so, that was the fundamental seed of that architecture, a forerunner of programmable ASIC (PASIC) products. And I defined two of those products. One was fully combinatorial (PLHS501). The other one was sequential, had registers on chip (PLHS502). But again that was using a bipolar oxide isolation process. I wanted to use CMOS but we did not have that at the time, we did not have the capability to do that. I knew, however, that in bipolar that product would not succeed very well and I was hoping that I could migrate to CMOS later. And that's when things turned for me again because I didn't like the way the Division was being run and the investment that we were making was minimal. There was some internal pressure and I decided to leave, and ironically I ended up going to MMI.

**Haines:** Well it was the logical place for programmable logic at the time.

**Cavlan:** Yes, yes. Yes, it was a logical place to be but in the end after I went in, it turned out it was not the wisest transition for me at the time because of several issues. One was fundamental based on technology capability internally at MMI at the time, and also the mindset. There was powerful thinking in the organization, and trying to change course into something different than that was not a very easy thing to do, even though the environment at MMI was much, much better in terms of dedication to programmable products because that was essentially all they had. I remember folding these elements together in terms of the moment. This is at a particular point for me because when I — first of all, after I developed PML I patented it. I wrote a paper and patented it the same way I patented IFL-20. I left Signetics before I could deliver the paper at Wescon that year, and actually wasn't in the room where they were presenting the paper. Someone else presented it. That wasn't a high point in my life… talk about an experience like that! But how did it happen that I went to MMI? That's an interesting story, yeah. I left Signetics for the second time, this time to go to MMI. What happened is that towards the end of '85, we're internally struggling at Signetics to find the next direction for us. Again I became very frustrated. One of the things that our Division manager wanted to do is to see if we also could get into PAL, second sourcing PAL, but our efforts by the Marketing group were going nowhere. So, one day I'm at my desk and leaving for home. It was about five o'clock. The phone rings and I thought, "Okay, I'll take it, the last call of the day". I grabbed the phone and the voice on the other side of the line was John Birkner's saying just like this, "Hey, hi Napoleone. It's John here." He said, "Do you know all that stuff we used to say about your programmable logic? That was all bull." He said, "Why don't you and I find a way to get together and develop a few products together?" I was speechless. I almost fell off the chair. I

  

said, “John, is that really you?” He said, “Oh, yeah.” I said, “Okay, well let’s get together.” Soon after that I invited him to come to my house for dinner and he came with his wife. Now, one thing that had happened prior to the story of me leaving Signetics is that, I don’t know if you <turning to Cline> recall, but you must recall, I think it was in the 1982 time frame, John Birkner and H.T. Chua, the original developers of the PAL architecture, got well recognized by MMI. They both got cars, company cars, and John got a Porsche. This was very well publicized in the press and John put a tag on it that spelled PAL JB. He went around the valley with that. A year later in 1983, I too got a car from Signetics. I got a Mercedes 380 SL, and so not to be undone, I promptly put on it the tag NC IFL. In retrospect, I regret that because it was kind of a childish upmanship. In fact, I even have it here. I’ll show you the tag. I saved it. See it’s right here <showing a license plate>. <laughter> Anyway, we went around with these two cars. So, when John called and I invited him for dinner, he came with his car and mine was parked in my driveway. So, he just rolls around with his Porsche and parks it right next to mine. I’m looking at these two cars and I’m saying, “Wow”. I am musing in my head, “What are the chances of something like that to ever happen?” In a way, I was thinking it reminded me of a scene of a movie where these two warring gunfighters just dropped their guns and decided to go have a friendly drink at the bar instead. <laughter> So, we had a nice evening. I cooked a nice meal for the evening and then later, after that successful encounter, we started having some dialogue inside our companies between management representatives, and that went nowhere. It was very difficult to have two guys like us try to collaborate on something new. At that time, if you remember, I’m talking about the 1985 time frame, mega PALs had come out and were not a success for MMI because of intrinsic issues with the product related to power and architecture. And so they, too, were really trying to engage in a product collaboration with us, which then dissipated. The way it worked out was that one day John and I went to lunch at the Lion & Compass in Sunnyvale, and he said, “You know, why do we waste our time doing this? Why don’t you just come over and join MMI?” So that is the story of how I went there.

[Going back, then, to my MMI experience, I tried to steer new PAL product development towards architectures including programmable state machines and PML structures. A couple of designs were started, but it didn’t work out. It was clear that MMI, reeling from the Mega Pal stumble, had lost its footing in the market place, and was struggling with three seemingly intractable problems: no plausible evolutionary or revolutionary products in the pipeline, no advanced CMOS and Bipolar processes in house, and besieged by several new competitors assailing directly or indirectly its established socket base in the market with faster, denser, and erasable versions of its established PAL line. A couple of months after my joining, John Birkner left the company. An attempt to partner with Xilinx floundered, and a company-wide strategy review with the consulting firm of Bain & Co. recommended for MMI to either acquire or be acquired. Soon after, MMI’s president, Irwin Federman, met with Jerry Sanders of AMD and sold the company to him. About a year later, I left AMD.]

**Haines:** So, Ron, do you have any thoughts that lead you, final thoughts or observations on this whole story?

**Cline:** Looking at the complications or the difficulties of bringing out a follow-on family, I think it’s one thing to be able to, almost on a skunk works basis, bring out a product that tests the market. In a way it’s an experiment and so by nature you want to make it simple, generic, easy to use, let customers explore what its capabilities are and I think that’s still a valuable thing to do. And you can do that with a very small team and, in fact, it should be done with a very small team because the idea is you want to put something out there cheaply and then let the market respond to what they want you to do next. And then whether I go to Tectronix or any other customer gives feedback and you start getting a framework of what

  

you want to do next, invariably it's something more complex and so you end up with this give and take with technology, marketing, manufacturing, cost, all of these things and trying to decide and then actually implementing that. I think the lesson and I've seen this many times is that you cannot do that follow-on more complex product with the same scale of development and resources as you did on the first product. And I think the company expectation, and this is just the nature I think of organizations, is they expect the same team to be able to accomplish more miracles in the same way they did before more complex and with more complexity. And dealing with complexity of products is the biggest challenge of engineering and business organizations. That does require a process. It does require organization structure and thinking that does take into account that complexity. If you hope that somehow engineers, key engineers or whatever specific talent is somehow going to drive through that complexity and end up with a successful follow-on product, no, it's not necessarily true. You really do have to deal with that complexity from an organizational point of view and we didn't know that or at least the higher up managers didn't really understand that.

**Cavlan:** Well I agree with that. In fact, one of the things, for example, that I think we could have done better, especially at the very beginning when the first FPLA came out — remember <turning to Cline> when I mentioned the fact that we were trying to address both the bit slice and byte organized applications? — was to do a better market segmentation to see where the filet of the market really was.

**Cline:** Yeah right. Yeah that's right.

**Cavlan:** And if we had done that I think we would have come up with a different structure, maybe even then a smaller device.

**Cline:** So a second round of market exploration, yeah.

**Cavlan:** Right, but there wasn't that you see, and I was in applications, so my mindset was not in that area, in the arena doing market study.

**Cline:** We had no market research.

**Cavlan:** So, you say just how do you do that? We ended up in a way just like building piper cub models in comparison to the jets they have today. And in response to what was brought about before, about the field programmable gate arrays being the first element of the architecture, of the hierarchical structure of programmable logic, you probably could designate those as being the wings of Icarus in terms of aviation, an aviation model. <laughter> So, it was that kind of thing, yeah… we should have done better market research, earlier development of design tools, software tools, because those would have been the roots of our product development. Once you know really the bulk of the market in that area, what their design style is — rather than being memory oriented it is more logic equation oriented — then you'd see, you're forced to see… and then you better do that.

**Cline:** So that's an area I think the industry in general has learned is to look at the products from the customer's point of view.

  

**Cavlan:** Yeah right.

**Cline:** And if marketing and management had gotten into that more and realized that they needed to put more resource in the development tools that would have made a big change.

**Cavlan:** But I think that the spirit that we had then is not here today. I mean, even though these large organizational structures have been created internally to assess better the risk of these adventures, that does not negate skunk works, and I don't think they should be discouraged because, yeah, you could have 100 failures but out of that if one success comes, and it is really the next thing in the marketplace that helps us, in general, to enhance it socially then, yes, that's the way we want to do that. And then in terms of lessons learned, additional lessons that I learned…one thing I could speak for myself is, "Don't break the designer's pick and shovel." I mean, he only has certain tools to do his job and if you go there and you put too much of a burden on him you're going to overwhelm his tools and he will fail at his job. The other thing is, you got to think of the product software development (as I wish we had done differently), that is "Don't just target a customer's socket." Just equally target his mindset, so that it's like you teach him how to play an instrument. Once he learns that's how he's going to make music, he's not going to be easily lured in trying to learn another one. The other thing too is, "If you're hitching your horse to the wrong cart, perseverance is no virtue." <laughter>

**Cline:** Is that Italian?

**Cavlan:** Well it's a little bit of philosophy but I don't want to just finish on a negative note.

**Haines:** Well let's finish this way. Are there any memories, special memories that really stick out during this, particularly the FPLF development during that time frame, things that for some reason really stand out for you?

**Cavlan:** Well from a development plan point of view, I remember distinctly an area where we had a lot of problems in the field. There was one time when after a couple years that the FPLS, the 28-pin that Ron had designed came out, we discovered that there was a race problem within the part that caused the state machine to just randomly jump into different states as it cycled around. This caused field failures. People that had already delivered equipment in the field said, "It just doesn't work." So, I was called in to go and troubleshoot this and find out what was happening, and finally after a lot of testing the part in various ways I identified what the problem was. It was essentially an internal speed mismatch that occurred because of process variations where Ron designed the Set-Reset Flip-Flops of the State and Output registers. They were not really edge triggered but gated clock Flip-Flops. As long as the clock was enabled, data would flow through the register. Now what happened was that primarily the internal State register jumped to random next-states. There were some OR array paths for the next-state that were very lightly loaded. They were feeding very few bits internally and so it was a fast path for data to flow through, come back around and go in the register again while the clock was still enabled; that was a big problem. This is how we salvaged the situation. To avoid the problem for parts that were already delivered I found a way to load… to look at the program table, look where the light loads were, and make sure that customers would not delete unused fuses… to keep them intact and present a higher capacitive load. And then, as a permanent modification, [I remember better now], I added a disabling input buffer driving all P-Terms across the AND array that was fed by the inverted phase of the clock. So, when the

  

clock came that line would be enabled after a small delay and would shut off the entire AND array after the positive clock phase had already jammed present-state data in the Flip-Flops. So, by the time the ensuing next-state data came around, the AND array was already disabled and the data couldn't go through to change the register contents. The race was fixed with an easy solution for this type of problem. There was another time when I — and this is another funny one — I went to England in 1979 to demonstrate the new FPLS. It was just introduced and so I went around the circuit for a talk. I visited several customers. I brought with me a Tic-Tac-Toe game that you could play with its FPLS program, lights on it and push buttons. That box was fed through an AC line and also had a battery back-up. So, when I went around various countries I used the battery mostly because I was sick and tired of switching plug adapters going from customer to customer. Eventually the battery discharged, and that came at a time when I was delivering this talk at the IE East Electrical Engineers at the Savoy Place in London, which you probably should know back there, Stephen <pointing to Stephen Smith, in the room>. It's by the Waterloo Bridge. So, I went into this building and when you go in there you see the statue of Faraday right in the front. It's very impressive. Then you go inside the main hall and you see gold characters on the walls around spelling the names of these great minds of the past like Newton, Maxwell, Kelvin and I said, "Gee what am I doing here? I'm not even worthy to lick the boots of such great men of science." But anyway I went in and gave my presentation. At a critical point when I was saying, "Okay now we have an FPLS and here's the box with the Tic-Tac-Toe" I invited a customer to play. He started pushing the buttons and nothing happens. And here I am. I said, "Oh, excuse me." Now I push the buttons, nothing happens. So, I started shaking and banging the box. Everybody in the audience started to get uncomfortable. At one point I guessed about the battery and said, "Okay, wait a minute gentlemen." I said, "Look, this is no accident and no surprise. We are sitting right by the Waterloo Bridge. My name is Napoleone. What else could you expect?" Now, in a British environment that's a tankful. Everybody started a big laughter, and I took that occasion to go in my bag where I had an adapter, plugged the box in the wall and made it work. So, that was a good recovery, Ah! <laughter>

**Haines:** Ron, what about you any thoughts like that about special moments that stand out?

**Cline:** Oh, that was just an exciting time. The FPLA was important. We were working on a lot of different things at the same time, didn't quite realize just that it was the first of something that it had a depth so I didn't really understand the excitement. That year was the year also that I courted and married my wife so it was just putting things in perspective. Nineteen seventy-five was a wonderful year.

**Haines:** Sure. So in thinking about where the technology might go in the next ten or 20 years, from your perspective with this technology, what do you think might happen?

**Cline:** I want to go back to the theme that I mentioned sort of at the beginning was that this was an opportunity of a certain technology capability meeting a need that was almost unknown at the time. I think the analog or a parallel I would use would be Clayton Christensen's book of meeting the demands of the market that you perceive from your customer base and at the same time a new technology comes along. Who would have thought in our time of the $25 part that programmable logic would be the low cost answer to systems and it ended up being that way of a brand new technology that just changes rules of the game. That's something that happens over and over again where something that looks like it's an outlier requires a skunk works project actually ends up becoming the basis of, a totally different …, a change in the paradigm for everything. So I would look again to what are the fundamental changes in technology that are coming along and how will they change things? And the way that I see that

  

happening then is the one thing that's been consistent from the early '70s all the way to current day, we put two billion transistors on a chip now but to first order we count on perfect transistors. We count on zero defects on a chip and that is a constraint that will not continue. I think that that will fundamentally change things in a way that makes programmable, what I'll call programmable hardware, not programmable logic but programmable hardware to be even more important in the future because the trend will be the need to handle imperfections and that requires adaptability, self programmability to be able to get around these defects which will fundamentally also mean that every chip is going to be different than all the other chips. You aren't going to be able to say, "I'm going to program this chip to do a certain thing, a certain thing." You're going to ask it to be able to do something but in the end it's going to have to figure out in its programmable hardware how it will accomplish that goal. I believe that my sense is that at some point probably within the next ten to 20 years that there will be hardware technology, programmable hardware technology that will end up becoming the mainstream solution of that particular problem so that would be my prediction.

**Haines:** So any last thoughts, any further thoughts then on any more stories before we close?

**Cavlan:** I want just to tell you one last thing in connection with the market success of programmable logic. After I left Signetics for MMI, the programmable logic story didn't finish at Signetics because, believe it or not, I went back to Signetics again in 1991 for a short period of time, and by 1991 Signetics' programmable logic book had grown considerably. You can see <showing a data book> it's a thick book. There are more than two-dozen parts in it, and contains still the products that I defined, plus others. It contains some PAL products, some ECL products, some CMOS products, especially some PML products. There are my original PML products plus a further version created in CMOS. So, my last path was undertaken in that area. Six years later it looks to me like this product group enjoyed vibrant sales for several years until finally things changed at Signetics totally in a different direction, and you <turning to Cline> went there again, I suppose, as well, right? You left for a while and then you went back.

**Cline:** And came back in the '90's yeah.

**Cavlan:** Yeah.

**Haines:** Gentlemen, thank you very much for your discussion and your stories today. This has been a great contribution to programmable logic history. I want to thank you, and on behalf of the Computer History Museum I thank you very much.

**Cavlan:** Thank you very much for the great job that you guys have done. It was really nice.

**Haines:** My pleasure.

**Cavlan:** Thank you very much.

END OF INTERVIEW

  